Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Aljaž Srša

Ogrodja za razvoj PHP aplikacij

diplomsko delo

univerzitetni študijski program prve stopnje
računalništvo in informatika

Ljubljana, 2016

Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Aljaž Srša

Ogrodja za razvoj PHP aplikacij

diplomsko delo

univerzitetni študijski program prve stopnje
računalništvo in informatika

Mentor: prof. dr. Viljan Mahnič

Ljubljana, 2016

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:
Ogrodja za razvoj PHP aplikacij

Tematika naloge:

Opišite najbolj razširjena ogrodja za razvoj PHP aplikacij, definirajte kriterije za njihovo medsebojno primerjavo in jih na podlagi teh kriterijev ovrednotite. Najbolje ocenjeno ogrodje nato uporabite za razvoj vzorčne spletne aplikacije. Aplikacijo predstavite tako z uporabniškega/ funkcionalnega kot s tehničnega vidika in na konkretnih primerih prikažite rešitve, ki jih izbrano ogrodje nudi.

Zahvaljujem se mentorju prof. dr. Viljanu Mahniču za sprejem diplomske teme ter vodenje in pomoč pri izdelavi te diplomske naloge.

Kazalo

Slike

Seznam uporabljenih kratic

kratica	angleško	slovensko
HTTP	HyperText Transfer Protocol	Protokol za prenos podatkov na spletu
HTML	HyperText Markup Language	Jezik za označevanje besedila namenjen za izdelavo spletnih strani
PHP	PHP: Hypertext Preprocessor	Odprtokodni programski jezik namenjen za strežniško uporabo
CSS	Cascading Style Sheets	Kaskadne stilske predloge
SQL	Structured Query Language	Strukturirani povpraševalni jezik za delo s podatkovnimi bazami
ORM	Object-relational mapping	Preslikava med objekti in relacijami
IRC	Internet Relay Chat	Internetni klepet

Povzetek

Naslov: Ogrodja za razvoj PHP aplikacij

V diplomskem delu so predstavljena štiri najbolj priljubljena ogrodja za razvoj spletnih aplikacij v programskem jeziku PHP: Laravel, Symfony, CodeIgniter in CakePHP. Ta ogrodja so primerjana med seboj glede na štiri kriterije, ki služijo v pomoč pri izbiri najustreznejšega. Ti so velikost skupnosti, kakovost uradne podpore, razumljivost dokumentacije in implementacija funkcionalnosti v posameznih ogrodjih, kjer je predstavljeno avtomatsko generiranje izvorne kode, usmerjanje, preslikava objektov v relacije in implementacija procesorja predlog. Zatem je predstavljena obrazložitev izbranega spletnega ogrodja, ki je uporabljeno v drugem delu diplomskega dela. V tem delu sta opisana izbira in potek razvoja spletne aplikacije, katera je razvita s pomočjo ogrodja, ki je bilo izbrano v prejšnjem delu. V razvoju spletne aplikacije so predstavljene in tehnično opisane ključne funkcionalnosti, kot sta na primer avtentikacija in avtorizacija. Na koncu je še prikazan potek končne spletne aplikacije.

Ključne besede: PHP, spletno ogrodje, Laravel, Symfony, CodeIgniter, CakePHP.

Abstract

Title: PHP frameworks

The thesis presents one of the four most popular PHP web frameworks: Laravel, Symfony, CodeIgniter and CakePHP. These frameworks are compared with each other according to the four criteria, which can help with the selection of a framework. These criteria are size of the community, quality of official support, comprehensibility of framework's documentation and implementation of functionalities in individual frameworks, which are automatic code generation, routing, object-relational mapping and implementation of template engines. Afterwards the selection of the framework is described, which is then used to develop a web application. Key functionalities of this web application, such as authentication and authorization, are presented and technically described. Finally, the usage of the developed web application is shown and illustrated.

Keywords: PHP, web framework, Laravel, Symfony, CodeIgniter, CakePHP.

Poglavje 1

Uvod

Dandanes je uporaba spletnih aplikacij široko razširjena, saj za delovanje teh aplikacij uporabniki potrebujejo le spletni brskalnik in povezavo z medmrežjem, s čimer se izognemo ogromni distribuciji in namestitvam, če bi namesto spletne aplikacije uporabili navadno. Spletne aplikacije so še posebej primerne za razne projekte, kot so na primer spletne trgovine, družbena omrežja in spletni portali.

Za razvoj take spletne aplikacije je na voljo veliko programskih jezikov, kot so na primer JavaScript, Ruby in Python, vendar smo se za to diplomsko delo odločili za uporabo programskega jezika PHP. Ta se izvaja na strani strežnika in je eden izmed najbolj priljubljenih jezikov za razvoj spletnih aplikacij. Po podatkih na strani W3Techs [7], je bil PHP v juliju leta 2016 uporabljen na kar 82,1% spletnih straneh, ki uporabljajo programske jezike na strani strežnika.

Ker je razvoj novih spletnih aplikacij časovno potraten, so se pojavila spletna ogrodja, ki skušajo razvijalcem znatno zmanjšati čas razvoja. To dosežejo tako, da implementirajo rešitve za pogoste komponente uporabljene v spletnih aplikacijah, kot so na primer povezava s podatkovno bazo, varnost, hitro postavljanje skeleta in ustvarjanje preprostih spletnih naslovov. Ker smo se odločili za uporabo programskega jezika PHP, se bomo osredotočili na primerjavo in uporabo PHP spletnih ogrodij.

V drugem poglavju diplomskega dela bomo najprej predstavili in opisali štiri priljubljena PHP spletna ogrodja Laravel, Symfony, CodeIgniter in CakePHP. Ta ogrodja nato med seboj primerjamo glede na kriterije, ki so predstavljeni v tretjem poglavju. V zaključku tega poglavja je utemeljena izbira ogrodja, ki se uporabi v naslednjih dveh poglavjih. V četrtem poglavju je opisana izbira spletne aplikacije in njenih ključnih funkcionalnosti. Zatem je v petem poglavju predstavljen razvoj in tehnična realizacija te spletne aplikacije, prav tako pa so opisane uporabljene tehnologije in orodja.

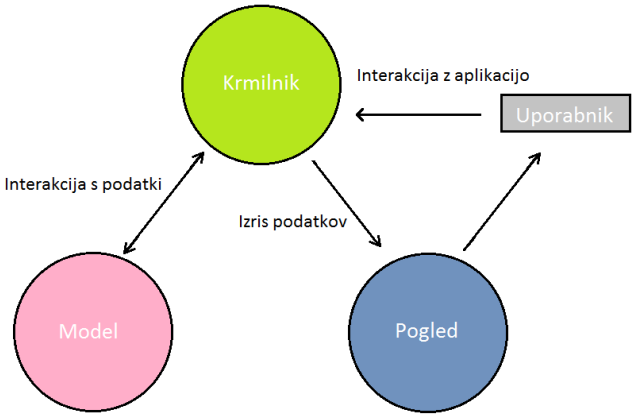
Poglavje 2

Spletna ogrodja PHP

Proces izdelovanja spletnih aplikacij je časovno potraten in zapleten, zato se za razvoj uporabi eno izmed spletnih ogrodij, ki so na voljo v večini brezplačno. Glavna naloga spletnega ogrodja je lajšanje dela, ki se pogosto pojavi pri razvoju novih spletnih aplikacij. Ogrodja večinoma podpirajo preprost dostop in upravljanje s podatkovno bazo, konfiguracijo URL poti, upravljanje seje in varnost.

Večina teh ogrodij deluje po principu MVC (angl. Model-View-Controller). Princip MVC razdeli spletno aplikacijo na tri dele: model, pogled in krmilnik. Model deluje na najnižjem sloju aplikacije in je zadolžen za upravljanje podatkovne baze in logiko aplikacije. Uporabnik nima neposrednega dostopa do modela, vendar pa lahko do njega dostopa preko pogleda ali krmilnika. Pogled je izpis kakršnihkoli informacij na spletno stran, katere dobi iz modela preko krmilnika, krmilnik pa skrbi za komunikacijo med uporabnikom in modelom. Enostavno MVC implementacijo lahko vidimo na sliki 2.1.

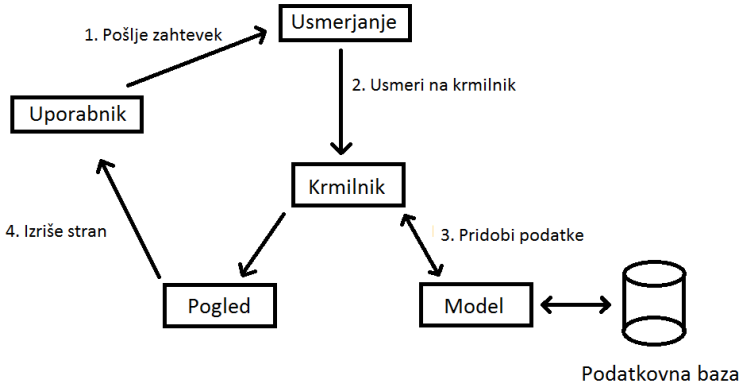
V tem diplomskem delu smo si zadali primerjavo PHP spletnih ogrodij, katera smo izbrali glede na priljubljenost med profesionalnimi in domačimi projekti. V nadaljevanju bomo predstavili štiri široko razširjena in popularna ogrodja: Laravel, Symfony, CodeIgniter in CakePHP. Ta ogrodja smo izbrali s pomočjo analize na spletu [8], kjer je prikazana popularnost ogrodij na spletni strani GitHub za leto 2015.



Slika 2.1: Preprosta MVC arhitektura.

2.1 Ogrodje Laravel

Spletno ogrodje Laravel [1] je izšlo junija leta 2011 in se zdaj smatra za eno izmed najboljših PHP ogrodij trenutno na trgu. V prvotni izdaji Laravel 1 je bilo pomanjkanje podpore krmilnikov, zato ogrodje ni ustrezalo kriterijem pravega MVC ogrodja. V kasnejši izdaji se je podpora krmilnikom dodala in se zdaj Laravel smatra za pravo MVC ogrodje. Namesto da bi Laravel na novo začel izumljati toplo vodo, si je izposodil veliko komponent iz obstoječega ogrodja Symfony, kot na primer komponento *HTTPFoundation*, usmerjanje, Carbon za boljše izražanje datuma in časa in močno Symfony komponento za ukazno vrstico (poimenovana Artisan v ogrodju Laravel). Vse te komponente se preprosto posodablja in upravlja s pomočjo programa Composer. Zato ker Laravel lahko vključuje veliko različnih neodvisnih komponent, se ta program uporablja zelo pogosto za dodajanje in posodabljanje novih komponent. Preprost primer delovanja aplikacije je razviden na sliki 2.2.



Slika 2.2: Preprost primer delovanja aplikacije Laravel.

2.1.1 Glavne funkcionalnosti

Laravel z namestitvijo vsebuje kopico pomembnih in uporabnih komponent. Najpomembnejše od teh so:

1. **Artisan** je močno orodje za ustvarjanje kode in upravljanje različnih nalog, kot so na primer vrste (angl. queues). Artisan uporablja komponento Symfony Console, katero se upravlja in posodablja preko orodja Composer. Artisan ima nabor ukazov, kateri pospešijo in poenostavijo deklaracijo raznih razredov. Moč je tudi definirati ukaze po lastni meri. Tu lahko vidimo primer ukaza za izdelavo razreda *model*:

```
php artisan make:model ImeRazreda
```

2. **Usmerjanje** (angl. routing) v Laravelu je fleksibilno in omogoča vezanje poti na katerikoli HTTP zahtevek. To je še posebej pomebno za ustvarjanje RESTful aplikacij. Najbolj uporabni HTTP zahtevki pa so GET, POST in PUT. Smeri lahko postavimo v skupine, na katere

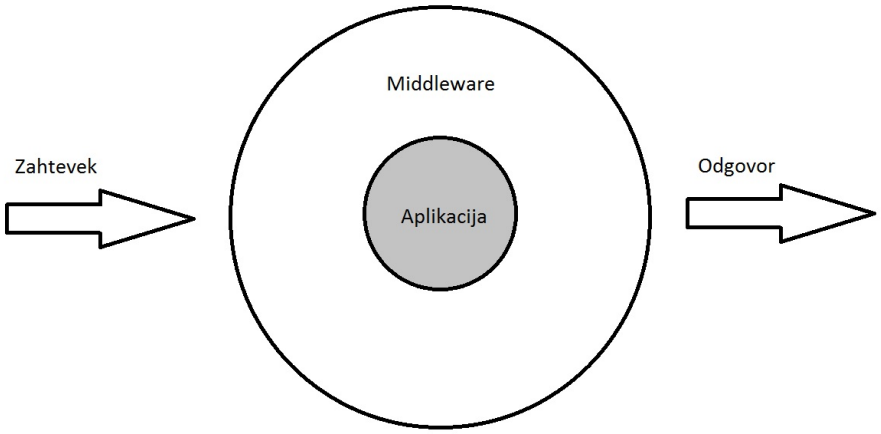
se pogosto definira imenske prostore ali vmesne mehanizme ogrodja Laravel (angl. middleware).

3. **Migracija** je način preprostega izdelovanja podatkovne sheme v programskem jeziku PHP. Omogoča lažje pisanje, prenos in nadzor podatkovne baze. Migracija vsebuje dve metodi. Prva metoda je poimenovana "up" in vsebuje kodo za izdelavo podatkove sheme. Druga metoda se imenuje "down" in naredi povsem nasprotno operacijo kot prva metoda. Da migracijo prenesemo, je potrebno uporabiti orodje Artisan in s tem migracijo naložimo na podatkovno bazo.

4. **Query builder** omogoča preprosto povpraševanje podatkovne baze v jeziku PHP. Namesto da se izgublja čas na dolgih in kompleksnih SQL stavkih, se uporabi metode iz fasade DB, katere je možno verižiti.

5. **Avtentikacija** se vzpostavi hitro in preprosto. V ukazni vrstici uporabimo orodje Artisan in s pomočjo ukaza *php artisan make:auth* postavimo (angl. scaffold) krmilnik in nekaj pogledov, potrebnih za osnovno prijavo v aplikacijo. Novo izdelani krmilnik vsebuje logiko za prijavo, katero se da prilagoditi na lastne potrebe.

6. **Vmesni mehanizem** (angl. middleware) je mehanizem filtriranja HTTP zahtev, ki vstopijo v aplikacijo. Najbolj osnoven primer vmesnega mehanizma je avtentikacijski vmesnik. Če uporabnik ni prijavljen v aplikacijo, ga vmesni mehanizem preusmeri na poljubni pogled. Vmesni mehanizmi se pogosto uporabljajo za avtentikacijo, avtorizacijo in beleženje zahtev. Preprosto delovanje vmesnega mehanizma lahko vidimo na sliki 2.3.

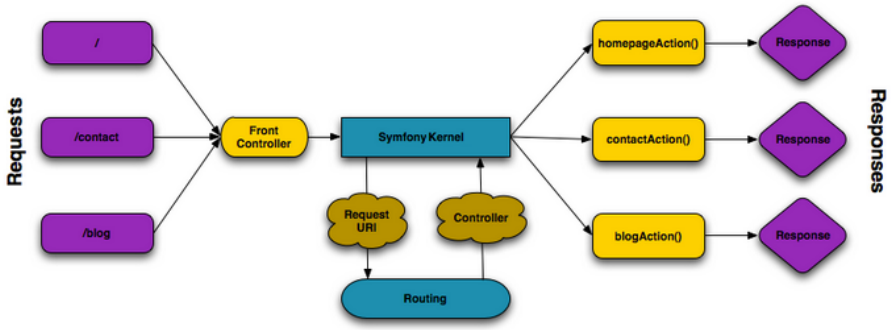


Slika 2.3: Delovanje vmesnega mehanizma v aplikaciji Laravel.

2.2 Ogrodje Symfony

Symfony [2] je odprtokodno spletno ogrodje, ki ga je ustvarilo francosko podjetje SensioLabs leta 2005. Ogrodje temelji na neodvisnosti in na večkratni uporabi komponent. Symfony tesno uporablja že obstoječe odprtokodne PHP projekte za nekatere osnovne lastnosti spletnega ogrodja. Vse komponente, ki jih Symfony ima na razpolago, so na voljo za uporabo v katerem-koli projektu kot neodvisne knjižnice. Nekateri znani projekti, ki temeljijo ali tesno uporabljajo Symfony komponente, so Laravel, Drupal in phpBB. Osnovna različica ogrodja Symfony ne vsebuje vseh komponent, vendar si jih razvijalec lahko prenese in namesti s programom Composer in si tako prilagodi okolje po svojih lastnih merah in potrebam.

Čeprav se Symfony smatra za MVC ogrodje, se njegov avtor s tem ne strinja močno. Avtor je mnenja, da je Symfony pravzaprav nabor komponent za večkratno uporabo. Kljub temu je to ogrodje postalo eno izmed najbolj sprejetih za uporabo v večjih podjetjih. Na sliki 2.4 je prikazan potek aplikacije v ogrodju Symfony. Ko aplikacija sprejme HTTP zahtevek, se glede na zapisani URL poišče pravo metodo, ki nato vrne HTTP odgovor.



Slika 2.4: Potek aplikacije Symfony [10].

2.2.1 Ključne komponente

Kot je bilo že omenjeno v prejšnjem odstavku, Symfony predstavlja nabor več neodvisnih komponent. Našteli bomo nekaj osnovnih in uporabnih komponent, ki so potrebne za razvoj aplikacije v temu ogrodju.

1. **HttpFoundation** je komponenta, ki objektno orientirano predstavi HTTP zahteve in odgovore. HTTP zahteva je predstavljena z objektom Request in vsebuje kopico informacij o poslani zahtevi. Nekatere od teh informacij so lahko podatki o piškotkih, poslani podatki z zahtevami POST ali GET in dostop do seje. Ta komponenta nam omogoča tudi ustvarjanje HTTP odgovora s pomočjo objekta Response.

2. **Komponenta Routing** se uporabi za mapiranje poti na določeno akcijo. Zapis teh poti oziroma smeri se lahko opravi na štiri načine: anotacije, YAML, XML ali PHP zapis. Za uporabo anotacij je potrebna namestitev komponente SensioFrameworkExtraBundle. YAML (YAML Ain't Markup Language) je način zapisa podatkov, ki je ljudem preprost za uporabo. Ta zapis se v ogrodju Symfony pojavlja v konfiguracijskih datotekah, zaradi hitrosti in preprostosti.

3. **Komponenta Form** omogoča izdelavo obrazcev kot objekta, kar pospeši in poenostavi izdelavo, ki se v spletnih aplikacijah pojavljajo ne-

prestano. Ta komponenta deluje tesno z nekaterimi ostalimi komponentami, kot na primer komponenta za validacijo podatkov in komponenta za ravnanje z zahtevki.

4. **Security** je komponenta, ki poskrbi za avtentikacijo, avtorizacijo in varnost pred CSRF (angl. Cross-Site Request Forgery) napadi. Če je potrebna avtentikacija uporabnikov shranjenih v podatkovni bazi, je potrebno namestiti dodatno komponento z imenom FOSUserBundle.

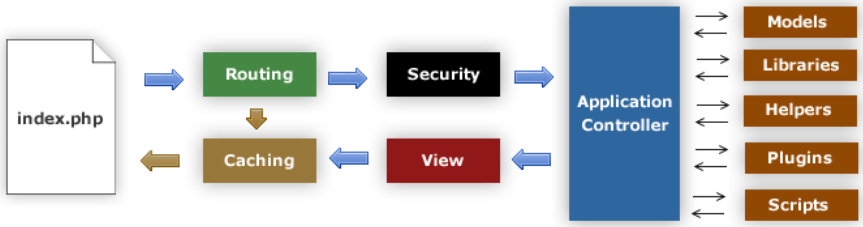
5. **Web Profiler in Web Debug Toolbar** sta orodji, ki omogočata pogled obsežnih informacij o spletni aplikaciji v razvoju. Primer take informacije je izpis vseh definiranih poti, varnost in splošne informacije o HTTP zahtevku.

2.3 Ogrodje CodeIgniter

CodeIgniter [3] je hitro in v primerjavi z drugimi izjemno enostavno spletno ogrodje, ki je na voljo že dobro desetletje. Cilji tega ogrodja so fleksibilnost, zmogljivosti in najhitrejša možna izvedba aplikacije v najmanjšem možnem paketu. Arhitektura ogrodja temelji na medsebojni neodvisnosti komponent, singularnost le-teh (neponovljivost komponent) in dinamično instanciranje komponent. Te se naložijo le takrat, ko so zahtevane.

Ogrodje ohlapno temelji na principu MVC, vendar nam omogoča tudi uporabo brez modelov, katere lahko združimo skupaj s krmilniki. CodeIgniter prav tako omogoča svobodo pri pisanju kode in ustvarjanju lastne arhitekture.

Primer poteka aplikacije z ogrodjem CodeIgniter je prikazan na sliki 2.5, kjer “index.php” naloži potrebne komponente za delovanje ogrodja. Ko CodeIgniter prejme HTTP zahtevek, gre ta skozi usmerjevalnik, ki poišče pravi krmilnik za izris potrebnega pogleda. Če je ta pogled že shranjen v predpomnilniku aplikacije, se iskanje in generiranje novega pogleda preskoči.



Slika 2.5: Potek aplikacije CodeIgniter [11]

2.3.1 Ključne prednosti

1. **Preprostost** je sinonim tega ogrodja, saj zelo enostavno dostopamo in začenjamo z njegovo uporabo. Ravno zaradi tega CodeIgniter blesti, zato je zelo primeren za začetnike, ki si želijo začeti uporabljati spletna ogrodja.

2. **Namestitev** ogrodja CodeIgniter je izjemno enostavna in hitra. Datoteke se preprosto prenese s spletne strani in naloži na strežnik. Potrebe po Composerju ali uporabe ukazne vrstice ni.

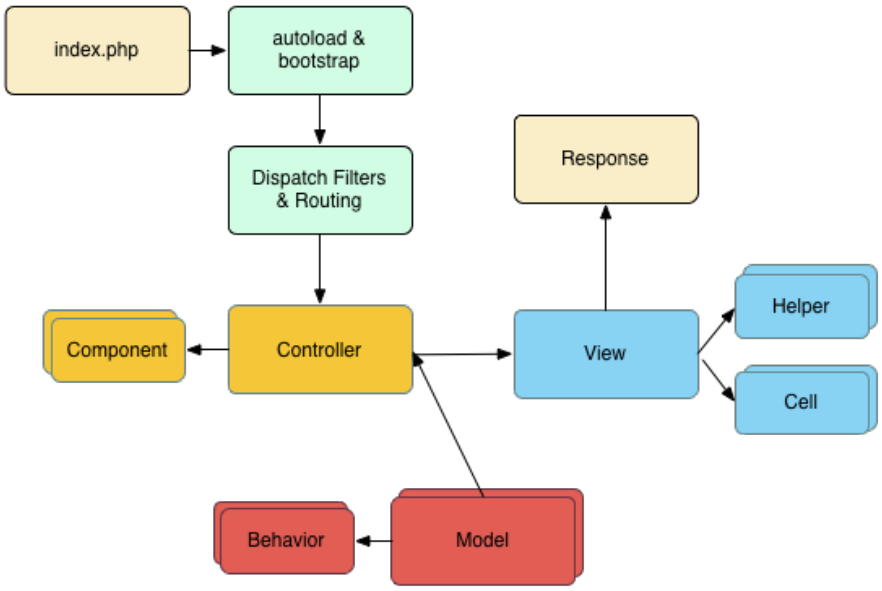
2.4 Ogrodje CakePHP

Ogrodje CakePHP [4] je razvil poljski razvijalec Michał Tatarynowicz leta 2005, navdih pa je pridobil iz spletnega ogrodja Ruby on Rails. Kljub temu, da je CakePHP eno izmed starejših PHP ogrodij na voljo, je na trgu še zmeraj popularno zahvaljujoč nekaterim principom, ki jih uporablja. Njihov najbolj izstopajoč princip je uporaba določenega zapisa kode oziroma "convention over configuration". Da se navadimo na ta določena pravila je potrebno nekaj časa, vendar ko ta pravila obvladamo, se konfiguriranje aplikacije drastično zmanjša. CakePHP deluje na principu MVC arhitekture, vendar razdeli model na dva dela:

- tabela, kjer so opisane razne relacije in validacije

- entiteta, kjer je predstavljen posamezen zapis v tabeli

Na sliki 2.6 je prikazan cikel življenja enega HTTP zahtevka.



Slika 2.6: Potek aplikacije CakePHP [12]

2.4.1 Ključne prednosti

1. **Bake konzola** je močno orodje, priloženo ogrodju, ki pomaga pri avtomatskem kreiranju in generiranju programske kode. Moč je ustvariti razne razrede, kot na primer entitete, krmilnike in poglede. Najbolj zanimiva funkcija pa je zagotovo generiranje CRUD (angl. Create, Read, Update and Delete) metod, ki ne le ustvari metode za te možne akcije, vendar ustvari tudi osnovne poglede oziroma strani.

2. **Konvencija** oziroma dogovor, je nabor pravil, kako zapisati in poimenovati razne razrede, spremenljivke in podobno. Ti dogovori so izjemno priročni, če si želimo prihraniti čas pri konfiguraciji spletne aplikacije. Kljub temu lahko dogovore zanemarimo, kar ni zaželjeno za delo v ekipi.

Poglavje 3

Kriteriji za izbor ogrodja

3.1 Uvod

V sklopu tega poglavja bomo predstavili kriterije, ki so po našem mnenju pomembni pri izboru spletnega ogrodja. Kriterij, s katerim bomo začeli, je velikost skupnosti in priljubljenost ogrodja. Bolj ko je ogrodje priljubljeno in razširjeno, večje so možnosti za razvoj novih vtičnikov (angl. plug-in) in razvijanje novih izdaj.

Sledi kriterij za dokumentacijo, ki je za razvijalce eden od pomembnejših kriterijev, saj lahko omogoča lažji pristop k uporabi spletnega ogrodja. V nasprotnem primeru lahko slaba in preobsežna dokumentacija oteži pristop in delo z ogrodjem.

Ker se lahko hrošči in težave pojavijo tudi v spletnih ogrodjih, je pomembna uradna podpora, ki skrbi za redno izdajo novih verzij s popravki.

Zadnji kriterij predstavlja ključne funkcionalnosti, ki najbolj pripomorejo pri izbiri ogrodja. Funkcionalnosti, ki smo jih izbrali so generiranje kode, usmerjanje, preslikava objektov v relacije in procesor predlog.

Na koncu tega poglavja smo izbrali spletno ogrodje glede na naštete kriterije in izbiro utemeljili.

3.2 Skupnost

Eden izmed najbolj pomembnih kriterijev ogrodja je velikost njihove popularnosti in skupnosti, saj le-to pomaga k hitrejšemu razvoju novejših verzij in kakovostnih komponent s strani skupnosti. Z večjo skupnostjo je tudi večja verjetnost, da se na spletu najde več virov in uporabnikov, ki z veseljem odgovorijo na razna vprašanja in pomagajo pri rešitvi težav. Velikost skupnosti smo primerjali na straneh GitHub, uradnih straneh in StackOverflow.

Laravel ponuja uradno stran za skupnost LaraCast, ki vsebuje kopico spletnih posnetkov za začetnike in tudi bolj izkušene razvijalce tega ogrodja. Večina teh posnetkov je na voljo brezplačno, vendar je za ostale posnetke potrebna mesečna naročnina. Poleg spletnih posnetkov je na voljo še forum z ogromno že obstoječih vprašanj in razprav. Prav tako je na spletu neuraden forum laravel.io, kjer se poleg standardnih razprav lahko pridružimo pogovoru v živo z uporabniki in ogledamo poddaje (angl. podcast), v katerih se predstavi razne novice.

Symfony so na svoji uradni strani imeli na voljo forum, vendar so ga pred enim letom zaprli, saj so mnenja, da je uporaba strani StackOverflow bolj koristna. Čeprav je forum uradno zaprt, je ogled starejših razprav možen, kar Symfony močno odsvetuje.

CodeIgniter na uradni strani vsebuje forum, ki izgleda precej aktivno na veliko različnih področjih. Prav tako je na voljo pogovor v živo z nekaterimi uporabniki, vendar je število le-teh v primerjavi z Laravelovim precej manjše.

CakePHP poleg standarnih forumov in IRC (Internet Relay Chat) pogovorov ponuja še uradno YouTube stran, kjer je trenutno naloženih več kot 160 posnetkov, večina katerih so posnetki z raznoraznih seminarjev.

Razen na uradnih straneh, se skupnosti večinoma združujejo na straneh GitHub in StackOverflow. V tabeli 3.1 je prikazana velikost priljubljenosti in število zastavljenih vprašanj na straneh GitHub in StackOverflow na dan 22. junija 2016.

Ogrodje	zvezdice na GitHub	vprašanja na StackOverflow
Laravel	23.931	40.558
Symfony	12.531	40.901
CodeIgniter	12.545	46.746
CakePHP	6.551	26.511

Tabela 3.1: Prikaz priljubljenosti na straneh GitHub in StackOverflow.

3.3 Dokumentacija

Verjetno je dokumentacija najpomembnejši kriterij, ki ga razvijalec zahteva od ogrodja. S slabo zapisano dokumentacijo je razvijalec prepuščen samemu sebi in pomoči s strani skupnosti, s presežno in prezahtevno dokumentacijo pa razvijalec porabi ogromno časa za obvladovanje ogrodja. Dobra dokumentacija mora biti primerna za začetnike in prav tako tudi v pomoč izkušenim uporabnikom.

Laravel nudi na svoji uradni strani povezavo do dokumentacije, kjer je opisan potek namestitve, osnove ogrodja, opis storitev in primer preproste ali zahtevne spletne aplikacije [5]. Poleg te dokumentacije nam je na voljo še API dokumentacija, v kateri so podrobno opisani vsi razredi in njihove metode.

Symfony ima na spletni strani dokumentacijo v obliki knjige, katera je na voljo tudi za prenos. Ta dokumentacija ozirom knjiga bi se nekaterim zdela prezapletena, saj je dolga več kot 200 strani. V knjigi so opisani temelji Symfonya, namestitev ogrodja, izdelava prve strani in pomembne osnovne komponente. Poleg knjige je na spletni strani obsežno število primerov in opis dobrih praks, ki so uporabne pri razvoju v tem ogrodju. Prav tako je na voljo API dokumentacija. Matično podjetje SensioLabs omogoča uradno Symfony usposabljanje ter pridobitev certifikata.

CodeIgniter dokumentacija je na voljo na njihovi uradni spletni strani. Vsebuje pregled ogrodja, postopek namestitve, preprost primer izdelave apli-

kacije in splošne teme. Na koncu so še reference za vse komponente, ki sestavljajo ogrodje.

CakePHP ima prav tako kot Symfony na razpolago dokumentacijo v obliki knjige [6], v kateri je predstavljeno ogrodje, namestitev in konfiguracija letega in osnovne teme oziroma komponente. Poleg knjige imamo na voljo še API dokumentacijo, kjer so opisani imenski prostori, razredi in metode.

3.4 Podpora

Ker se stvari v svetu tehnologije in računalništva nenehno spreminjajo in nadgrajujejo, je potrebno da so razvijalci z vsemi spremembami seznanjeni in nanje pravilno pripravljeni. To je še posebej pomembno pri razvijanju spletnih aplikacij, zato je pri izbiri spletnega ogrodja odločilna življenjska doba izdaje in podpora oziroma vzdrževanje s strani ogrodja. Od ogrodja je pričakovati, da se posodobi ali nadgradi redno. To je ponavadi zapisano v načrtu proces objavljanja (angl. release process), kjer je opisan cikel uradnih objav novih izdaj.

Laravel je najnovejše ogrodje od naštetih, zato je njihov načrt objavljanja začel veljati z zadnjo izdajo - 5.1. Avtor ogrodja je predpostavil, da se glavne izdaje objavljajo vsaki dve leti s tem pa je vključena podpora težavam (dve leti) in podpora varnostnim težavam (tri leta). Temu načinu se uradno reče "Long-term support" (LTS) in je prisotna pri objavah novih izdaj ogrodij. Laravel prav tako ponuja podporo manjšim različicam, katere se objavljajo vsakih šest mesecev in imajo podporo do enega leta.

Symfony ima podoben pristop kot Laravel. Manjše izdaje objavlja vsakih 6 mesecev in ponuja podporo do štirinajst mesecev. Nove izdaje, prav tako kot Laravel, objavlja vsaki dve leti, vendar nudi podporo do štirih let. Ta načrt je razviden na njihovi spletni strani [9].

Čeprav je CodeIgniter eno izmed starejših ogrodij, načrta za proces objavljanja ne uporablja. Prav tako podobnega načrta ne uporablja tudi CakePHP.

3.5 Funkcionalnosti

Funkcionalnosti ogrodja so verjetno najbolj odločilen kriterij za razvijalca na osebni ravni, saj te pripomorejo k hitrosti in preprostosti razvijanja. Seveda različna ogrodja vseh funkcionalnosti ne podpirajo, vendar se nekaterim te komponente lahko dodajo z lahkoto. V nadaljnjem bo naštetih in opisanih nekaj pomembnih komponent oziroma funkcionalnosti, ki so dandanes še izredno pomembni.

3.5.1 Generiranje izvorne kode

Generiranje kode je način avtomatskega ustvarjanja kode po neki predlogi, ki se v projektu pojavlja pogostoma (angl. boilerplate). Najbolj pogosto se generira razrede za krmilnike, modele, dogodke in teste. Ta proces razvijalcu poenostavi programiranje, saj se lahko osredotoči na logiko aplikacije in ne izgublja časa s trivialno kodo.

Laravel generiranje kode vključuje v projektu in jo izvaja preko ukazne vrstice s pomočjo konzole Artisan. S to konzolo je moč ustvariti kodo za pomembne Laravel razrede, kot na primer krmilnike, dogodke in njihove poslušalce in migracije za podatkovno bazo. Poleg generiranje razredov Artisan ponuja še postavitev osnovne spletne aplikacije. Ta vsebuje registracijo novih uporabnikov, prijava le-teh in implementacija potrebnih pogledov. S tem se razvoj aplikacije znatno zmanjša. Laravel nam poleg privzetih ukazov omogoča še ustvarjanje dodatnih ukazov po meri.

Na primeru spodaj lahko opazimo ustvarjanje razreda za dodatni ukaz.

```
php artisan make:console PosljiNovico --command=novice:poslji
```

Symfony orodje za avtomatsko ustvarjanje kode je vključeno v standardno različico ogrodja, vendar je, tako kot vse komponente, na voljo tudi kot samostojna komponenta. Omogoča nam kreacijo skeleta aplikacije, krmilnikov, podatkovnih entit, razredov za obrazce in nove ukaze. Komponento se uporablja preko ukazne vrstice, tako kot pri ogrodju Laravel.

CakePHP prav tako vsebuje generiranje kode preko ukazne vrstice, vendar omogoča kreiranje krmilnikov z osnovnimi CRUD (Create, read, update and delete) metodami, modelov in pogledov glede na podatke iz podatkovne baze. To dosežemo z ukazom: *bin/cake all*

Ogrodje CodeIgniter lastne možnosti generiranja kode ne podpira, vendar obstajajo neodvisne komponente, katere se lahko uporabi v ogrodju CodeIgniter.

3.5.2 Usmerjanje

Za usmerjanje v ogrodju skrbi usmerjevalnik (angl. router), ki poenostavi URL smer in omogoča preprost zapis povezave med URL smerjo in krmilnikom in njegovimi akcijami. Večinoma so te poti zapisane v konfiguracijskih datotekah, vendar je v nekaterih primerih možna tudi anotacija v krmilniku.

Smeri v ogrodju Laravel se zapiše s pomočjo razreda Route, ki nam omogoča ogromno priročnih funkcionalnosti. Smerem lahko dodajamo različne filtre oziroma vmesne mehanizme, različne HTTP zahtevke in definiramo imena. Smeri lahko prav tako umestimo v večje skupine, na katere lahko prav tako dodajamo filtre. Na naslednjem primeru je razvidna skupina smeri.

```
Route::group(['middleware' => 'web'], function () {
    Route::post('prijava', 'LoginController@izvediPrijavo');
    Route::get('dom', 'HomeController@index');
    Route::post('novice/ustvari','NewsController@ustvariNovico')
        ->middleware(['admin']);
    Route::get('novice/{id}', 'NewsController@pokaziNovico');
});
```

Symfony za zapis smeri uporablja štiri načine: anotacije, zapis YAML, zapis XML in zapis v PHP. Izmed teh štirih načinov je verjetno najbolj preprost zapis z anotacijami, vendar s tem načinom smeri niso centralizirane v eni konfiguracijski datoteki. Sledi primer zapisa smeri z anotacijo.

```
/**
 * @Route("/novica/{novicaId}", name="pokaziNovico")
 */
public function novicaAkcija($novicaId) {
  //jedro funkcije
}
```

Smeri v CodeIgniter usmerjevalniku so preproste PHP tabele in vsebuje dve rezervirani smeri za privzeti domači krmilnik in privzeti pogled za HTTP napako 404. Usmerjevalnik podpira uporabo HTTP zahtevkov za zapis smeri, kar je primerno za razvoj REST aplikacije. Sledi prikaz preprostega primera.

```
$route['novice/(:num)'] = 'novice/pokaziNovico/$1';
```

Usmerjevalnik pri ogrodju CakePHP deluje podobno kot pri ogrodju Laravel, z objektom Router. Omogoča nam zapis smeri po principu DRY (angl. Don't Repeat Yourself) s tem, da lahko smeri, ki so v istem imenskem obsegu (npr. /novice) definiramo v skupini oziroma obsegu. To lahko opazimo na slednjem primeru.

```
Router::scope('/novice', ['controller' => 'Novice'], function(
    RouteBuilder $routes) {
     $routes->connect('/poisci/:id',
          ['controller' => 'Novice', 'action' => 'poisciNovico'],
          ['_name' => 'namedRoute']
     );
});
```

Vsi usmerjevalniki v principu delujejo enako, a v vsakem ogrodju so te implementirani različno. Uporaba nekaterih usmerjevalnikov je bolj intuitivna in preprosta, kar vpliva na končno izbiro.

3.5.3 Preslikava med objekti in relacijami

Preslikava med objekti in relacijami (ORM) je plast med podatkovno bazo in aplikacijo ali metoda za pretvorbo podatkov v programsko razumljive objekte, kateri poenostavijo branje in manipuliranje podatkov. ORM orodja lahko uporabijo vzorec "active record", vzorec "data mapper" ali pa hibrid obeh vzorcev. Vzorec "active record" je eden izmed najbolj uporabljenih vzorcev in deluje tako, da je vsak zapis oziroma vrstica v podatkovni bazi predstavljena kot objekt razreda tabele. To omogoča lažje shranjevanje in uporabo podatkov, saj objekt podeduje vse metode osnovnega razreda, ki ima potrebne podatke za povezavo s podatkovno bazo. Objekti pri vzorcu "data mapper" pa so preprosti PHP razredi in teh informacij nimajo, zato za shranjevanje in posodabljanje podatkov potrebujejo entitetne upravitelje. Ta način delovanja je izjemno enostaven in hitrejši, vendar je uporaba bolj formalno stroga.

Laravel za implementacijo ORM uporablja komponento Eloquent, ki je implementacija vzorca "active record". Implementacija razredov se izvede s pomočjo konzole Artisan, ki poimenuje razred z imenom tabele v podatkovni bazi. Eloquent predpostavlja, da so primarni ključi poimenovani kot id, vendar je te predpostavke možno nastaviti po meri. Ker smo uporabili relacijsko podatkovno bazo, je potrebno te relacije posredovati Laravelu oziroma Eloquentu, kar je enostavno nastaviti. V modelu je potrebno definirati to relacijo kot PHP metodo, kar se vidi na temu primeru.

```
class Novica extends Model
{
  public function komentar() {
        return $this->hasMany('App\Komentar');
    }
}
```

Symfony v nasprotju z Laravelom implementira komponento Doctrine ORM, ki uporablja vzorec "data mapper". Relacije so definirane s pomočjo

anotacij v posameznem modelu. Ker Doctrine deluje po principu "data mapper", je za zapis v podatkovno bazo potreben upravitelj entitet. Ta postopek je razviden iz naslednjega primera.

```
$user = new User();
$user->setName('Ime');
$upravitelj = $this->getDoctrine()->getManager();
$upravitelj->persist($user);
$upravitelj->flush();
```

CodeIgniter nima implementirane ORM komponente, vendar je možna namestitev Eloquent, Doctrine ali katerikoli druge ORM komponente s pomočjo Composerja. Glede na to, da CodeIgniter ne uporablja preslikave med objekti in relacijami, se obdelovanje baze lahko opravi z razredom Query Builder. Na primeru spodaj lahko vidimo grajenje povpraševanja podatkovne baze.

```
$this->db->select('novice.*, uporabniki.ime');
$this->db->from('novice');
$this->db->join('uporabniki',
    'uporabniki.id = novice.uporabnik_id');
$query = $this->db->get();
```

Ogrodje CakePHP implementira lastno rešitev objekto-relacijskega mapiranja, ki je mešanica vzorcev "active record" in "data mapper" in je specializirana za uporabo z relacijskimi podatkovnimi bazami. ORM komponento je moč tudi uporabiti samostojno v drugih projektih.

3.5.4 Procesor predlog

Procesor oziroma razčlenjevalnik predlog je orodje za lažji razvoj čelnega dela aplikacije (angl. front-end). Omogoča enostavnejši dostop do spremenljivk iz krmilnika in v nekaterih primerih tudi enostavnejšo uporabo PHP funkcij. Čeprav je uporaba teh predlog čistejša in preprostejša, je uporaba čistega PHP-ja znatno hitrejša. Kljub temu so predloge na voljo v večini ogrodij.

Laravel uporablja razčlenjevalnik z imenom Blade in ima na razpolago veliko funkcij, katere je možno dodati po lastni meri. Za razliko od drugih razčlenjevalnikov, Blade ne omejuje uporabe PHP-ja v pogledih. Uporabna funkcionalnost procesorja Blade je uporaba odsekov v pogledih. S tem lahko ustvarimo glavni postavitveni pogled, katerega lahko uporabimo kot starševski pogled in nanj povežemo različne poglede. Blade ima na razpolago še mnogo uporabnih funkcij, ki se začnejo z znakom afna (@). Iz naslednjega izseka je videti primer predloge.

```
@foreach($novice as $novica)
  <article>
    <h4>{{ $novica->naslov }}</h4>
    <p>{{ $novica->telo }}</p>
  </article>
@endforeach
```

Symfony ima na voljo razčlenjevalnik Twig, ki je podoben Laravelovemu procesorju predlog. Za razliko od Blada, Twig ne omogoča uporabe jezika PHP v predlogah. Omogoča pa uporabo blokov in dodajanje lastnih funkcij. V primeru lahko opazimo uporabo bloka in nekaj funkcij.

```
{% block telo %}
  {% for uporabnik in uporabniki %}
    <li>{{ uporabnik.email|escape }}</li>
  {% endfor %}
{% endblock %}
```

Za razliko od prejšnjih dveh razčlenjevalnikov, CodeIgniter vsebuje le preprost procesor predlog. Možnost ima izpisa spremenljivk in preprosto zanko. Avtorji ogrodja CodeIgniter ne spodbujajo uporabe procesorja predlog, ker si želijo hitro ogrodje, kar bi uporaba predlog lahko omejila.

CakePHP uporablja svoj lasten preprost razčlenjevalnik predlog in uporablja alternativno PHP sintakso, kjer se uporaba oklepajev in zaklepajev zamenja z drugačno sintakso. Taka sintaksa je razvidna na naslednjem primeru preproste zanke, kjer funkcija h() spremeni niz v varnejši HTML izpis.

```
<?php foreach ($news as $news): ?>
  <tr>
    <td><?= h($news->title) ?></td>
    <td><?= h($news->body) ?></td>
  </tr>
<?php endforeach; ?>
```

Kljub preprostosti teh predlog, je možno dodati katerikoli razčlenjevalnik, kot na primer razčlenjevalnik predlog Twig.

3.6 Utemeljitev izbora

Na podlagi opisanih kriterijev je bilo možno narediti odločitev za izbiro spletnega ogrodja. Seveda se ta odločitev razlikuje med domačimi in profesionalnimi projekti, kjer na odločitev vpliva več različnih faktorjev. Najpomembnejši kriterij za profesionalne projekte je podpora s strani ponudnika ogrodja, kjer je očitna izbira Laravel ali Symfony, saj oba ogrodja ponujata dolgoročno podporo do vsaj dve leti. Glede na ponujeno podporo, ali bolje rečeno pomanjkanje podpore, je za domačo uporabo bolj primeren CodeIgniter ali CakePHP.

Dokumentacija se med naštetimi ogrodji razlikuje v obsežnosti in razumljivosti. Ta razlika je največja pri Symfony, saj ima na voljo dokumentacijo z veliko primeri, ki so morda malce preobsežni. Menimo, da ima Symfony

strmo krivuljo učenja in potrebuje največ časa in sredstev za nemoteno uporabo ogrodja. Dokumentacija ogrodij Laravel in CakePHP je obsežna in preprosta, vendar omogoča hiter prijem tehnik in arhitekture. Mnenja smo, da je dokumentacija ogrodja CodeIgniter nepregledna in je tudi edino ogrodje, ki ne ponuja API dokumentacije.

Mnogokrat se pripeti, da se pojavi hrošč ali napaka, za katero ni videti jasne rešitve. Tu nastopi podpora s strani skupnosti, kjer se najdejo rešitve in odgovori na razna vprašanja o ogrodju in njegovi prihodnosti. Trenutno najbolj priljubljena spletna stran za taka vprašanja je StackOverflow, kjer je moč vprašati za katerokoli programsko, tehnološko ali računalniško vprašanje. Od naštetih ogrodij verjetno najbolj izstopa CakePHP, ki ima na StackOverflow zastavljenih polovico manj vprašanj, kot ostala tri ogrodja. Navkljub temu, da je Laravel najmlajše ogrodje, ima hitro rastočo skupnost in je ena izmed največjih med PHP ogrodji. Na voljo imajo mnogo spletnih posnetkov za začetnike in tudi spletne oddaje, na katerih je gost pogostoma avtor ogrodja.

Funkcionalnosti ogrodja so izjemno pomemben kriterij ogrodja, saj poskušamo poiskati tako ogrodje, ki nam bo omogočalo doseči cilj z orodji in funkcionalnostmi, ki jih ponuja. Iz tabele 3.2 je razvidno, da ogrodje CodeIgniter nima implementiranih uporabnih funkcionalnosti, ali pa so te implementacije preproste.

CakePHP ima lastno implementacijo nekaterih funkcionalnosti, kot na primer komponento za objektno-relacijsko mapiranje in razčlenjevalnik predlog. Kljub temu je pomanjkanje uradne podpore in velikost skupnosti slabša stran tega ogrodja.

Končna odločitev je bila med ogrodjema Laravel in Symfony. Poleg dobre implementacije splošnih funkcionalnosti, imata oba ogrodja podoben načrt objavljanja novih verzij in dolgoročno podporo (LTS). V primerjavi ima Laravel manjšo krivuljo učenja, kar omogoča hitrejši oprijem in uporabu ogrodja. Hitro rastoča skupnost in priljubljenost ogrodja Laravel je bil odločilni kriterij za izbor le-tega.

Kriterij	**Laravel**	**Symfony**	**CodeIgniter**	**CakePHP**
Skupnost	Forum, spletni posnetki, IRC, spletne oddaje	Uradni forum premeščen na StackOverflow	Forum, IRC	Forum, YouTube kanal
Dokumentacija	Dokumentacija s primeri, API	Knjiga, API	Dokumentacija s primeri	Knjiga, API
Podpora	LTS	LTS	Ni znano	Ni znano
Generiranje kode	Ukazna vrstica	Ukazna vrstica	Ne podpira	Ukazna Vrstica
Usmerjanje	Objekt	YAML, anotacije	Tabela	Objekt
ORM	Eloquent, active record	Doctrine 2, data mapper	Ni implementirano	Lastna implementacija
Predloge	Blade	Twig	Preprost razčlenjevalnik	Lasten razčlenjevanik, alternativna php sintaksa

Tabela 3.2: Prikaz vseh kriterijev.

Poglavje 4

Opis spletne aplikacije

4.1 Izbor spletne aplikacije

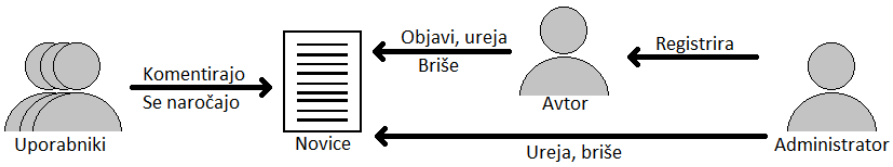
Po izbiri spletnega ogrodja, je bilo potrebno razmisliti o spletni aplikaciji, ki bo razvita z izbranim ogrodjem. Da bi prikazali odlike ogrodja Laravel, smo se odločili za razvoj spletnega portala za ogled in objavo novic. Dandanes so take spletne aplikacije redno obiskane, saj ponujajo sveže in aktualne novice, katere se razlikujejo od strani do strani. Nekateri spletni portali nad katerimi smo se zgledovali so:

- www.delo.si
- www.rtvslo.si
- www.bbc.co.uk
- www.forbes.com

Spletni portal za ogled in objavo novic je dober primer CRUD (ustvari, beri, posodobi, in izbriši) aplikacije, kjer se ustvarja in posodablja novice in komentarje. Za uporabo teh CRUD metod je pomembna implementacija prijave v sistem, kar je dober primer za uporabo spletnega ogrodja.

4.2 Funkcionalnosti aplikacije

Preden se prične tehnična realizacija spletne aplikacije, si je pomembno zastaviti nekaj ključnih funkcionalnosti te aplikacije. Enostaven potek portala je opisan na sliki 4.1.



Slika 4.1: Potek spletnega portala.

4.2.1 Prijava v sistem

Ker smo si za izdelavo izbrali spletni portal za objavo in ogled novic, je pomembna realizacija prijave v sistem. Za ta portal smo zastavili tri uporabniške vloge: administrator, avtor in uporabnik. Vloga administratorja je registrirati nove avtorje, možnost urejanja in brisanja katerihkoli novic ter brisanje uporabnikovih komentarjev. Ko administrator doda novega avtorja, ima ta možnost dodajanja novih novic preko portala. Če je potrebno, je avtorju omogočeno urejanje ali brisanje le svojih lastnih novic. Spletni uporabnik ima možnost ogleda novic in obstoječih komentarjev brez prijave, vendar je ta potrebna za dodajanje novih komentarjev.

4.2.2 Objava novic

Objava novih novic se izvede preko spletne aplikacije s pomočjo WYSIWYG (angl. What You See Is What You Get) urejevalnika besedil. To pomeni, da takrat, ko avtor izdeluje in oblikuje spletno novico, sproti vidi že končni izdelek, ki bo izrisan na spletni strani. Prav tako je ta princip uporabljen pri

urejanju teh besedil.

4.2.3 Komentiranje novic

Te novice so na voljo za ogled vsem obiskovalcem spletnega portala, vendar je komentiranje teh novic omogočeno le registriranim uporabnikom. Trenutno prijavljen uporabnik lahko pod ogledano novico ustvari nov komentar, ki je nato viden vsem obiskovalcem. Če s svojim komentarjem ni zadovoljen, ima uporabnik na voljo izbris tega komentarja. Prav tako lahko komentarje izbriše administrator portala, v primeru da komentar vsebuje neprimerno vsebino.

4.2.4 Naročanje na novice

Če uporabniki želijo prejemati sveže novice na svojo e-pošto, imajo možnost, da se naročijo na specifično kategorijo oziroma rubriko. Ko je v tej kategoriji dodana nova novica, se naročenim uporabnikom na njihov e-poštni naslov pošlje pošta s povezavo do te novice. Če se uporabnik želi odjaviti od prejemanja novic, lahko to naredi preprosto na istem mestu, kjer se je na novice naročil.

4.2.5 Podatki o vremenu

Vsi obiskovalci lahko na glavni spletni strani opazijo podatke o vremenu za lokacijo, odkoder dostopajo do spletne aplikacije. Ti podatki prikazujejo trenutno temperaturo ozračja in preprosto informacijo o stanju vremena. Pod tem je videti še podatke o naslednjih štirih dneh, kjer je opisano stanje vremena in temperatura ozračja zjutraj in popoldne.

4.3 Realizacija funkcionalnosti s pomočjo ogrodja

4.3.1 Povezava s podatkovno bazo

Prvi korak, ki je potreben pri realizaciji spletne aplikacije, je postavitev podatkovne baze. V ogrodju Laravel je ustvarjanje tabel možno brez uporabe samostojnih programov ali uporabe jezika SQL, saj se to lahko izvede s pomočjo migracij. To je preprost razred v katerem se nahajata metodi "up" in "down". V metodi up se s pomočjo objekta za gradnjo shem zapiše shemo tabele, katero se prenese na podatkovno bazo preko ukazne vrstice. Metoda down pa izvede obratne operacije kot v metodi up. Na primer, če v metodi up ustvarimo novo tabelo, bo metoda down to tabelo izbrisala. Ko so te migracije prenesene na podatkovno bazo, se nato ustvari model za vsako tabelo. V modele se poleg lastnih metod lahko zapiše še razne nastavitve in metode, ki spreminjajo vnos ali izpis podatkov. Za pravilno delovanje objektno-relacijskega mapiranja je potrebno določiti še razmerja z ostalimi tabelami oziroma modeli.

4.3.2 Prijava v sistem

Dandanes ima večina spletnih aplikacij implementirano prijavo v sistem in je nekaj, kar je potrebno razviti znova za vsako aplikacijo. Vendar Laravel ponuja hiter način postavitve (angl. scaffold) vseh smeri, krmilnikov, pogledov in migracije, ki so potrebni za osnovno registracijo in prijavo uporabnikov. Ta hitra postavitev skeleta se izvede preko ukazne vrstice in je priporočeno, da se to opravi le na svežih aplikacijah. Ker je vsaka aplikacija različna, je potrebno prilagoditi delovanje prijav in registracij. To ni težavna naloga, saj so vse metode dobro dokumentirane.

4.3.3 Ustvarjanje novic

Dodajanje novih novic je glavna naloga vloge avtorja, urejanje in brisanje novic pa je omogočena avtorju in administratorju. Za intuitivno oblikovanje

končne novice je bilo potrebno uporabiti WYSIWYG urejevalnika besedil, ki v ogrodju Laravel ni na voljo, vendar je implementacija takega urejevalnika preprosta. Ker se za novice spodobi, da vključujejo razne slikovne elemente, je bilo potrebno implementirati sistem za uporabo in prenos slikovnih datotek. Za to smo uporabili odprto kodno komponento za upravljanje datotek, ki ponuja preprosto implementacijo skupaj z urejevalnikom besedil. Namestitev teh komponent v ogrodje Laravel je enostavno izvesti s pomočjo programa Composer.

4.3.4 Naročanje na novice

Ker lahko pošiljanje le ene elektronske pošte vzame kar precej časa, lahko to povzroči zastoj spletne aplikacije posameznemu uporabniku in ker se naročene novice pošiljajo večim uporabnikom, bi bila uporaba take aplikacije neznosna. Zato je potrebno implementirati način pošiljanja elektronske pošte, ki se lahko izvede neodvisno od spletne aplikacije. To je v Laravelu možno enostavno izvesti s pomočjo čakalnih vrst (angl. queues), kjer se razne uvrščene naloge izvedejo v kasnejšem času. Za čakalne vrste se lahko uporabi Redis, Amazonove čakalne vrste (Amazon SQS) in Beanstalkd, vendar smo za razvoj naše spletne aplikacije uporabili implementacijo čakalnih vrst v podatkovni bazi. Z uporabo čakalnih vrst se izvajanje raznih nalog preseli v ozadje in s tem uporabniku znatno pohitrimo uporabo aplikacije.

Poglavje 5

Tehnična realizacija

Spletno aplikacijo in njene ključne funkcionalnosti, ki so bile opisane v prejšnem poglavju, je bilo potrebno realizirati s pomočjo izbranega ogrodja. S tehnično realizacijo prikažemo razvoj aplikacije z uporabo prej opisanih lastnostih in značilnosti ogrodja Laravel.

5.1 Opis uporabljenih tehnologij

Dandanes je za razvoj standardne spletne aplikacije potrebno uporabiti kopico tehnologij. Za razvoj pogledov oziroma strani se uporabi označevalni jezik HTML, oblikovanje teh HTML vsebin pa se opravi s kaskadnimi stilskimi predlogami (CSS). Ker se od novejših spletnih aplikacij pričakuje, da so na nek način interaktivne, je zaželjeno razvijanje s tehnologijo JavaScript, ki omogoča ustvarjanje interaktivnih spletnih strani. Razvoj v ogrodju Laravel pa seveda poteka v programskem jeziku PHP. Za delujočo spletno aplikacijo je prav tako pomembna podatkovna baza, katero obdelujemo s povpraševalnim jezikom SQL. V nadaljevanju bomo na kratko opisali te tehnologije.

5.1.1 PHP

PHP (včasih poimenovan Personal Home Page, trenutno PHP: Hypertext Preprocessor) je odprtokodni programski jezik, ki se primarno uporablja za

razvoj spletnih aplikacij in se izvaja na strani strežnika. Ta programski jezik je šibko tipiziran, kar lahko povzroči veliko napak začetnikom programiranja. Zaradi svoje preprostosti je PHP eden izmed najbolj priljubljenih tehnologij za razvoj spletnih aplikacij, vendar je s tem pridobil kar nekaj slabega ugleda. Kljub temu je ta programski jezik široko uporabljen, čemur močno pripomorejo PHP ogrodja. Del PHP kode je možno vdelati v HTML datoteke, uporabljena pa je lahko tudi kot samostojna koda, ki se izvede na strežniku s pomočjo interpreterja. Zgradba in sintaksa jezika je podobna jezikom C, C#, Perl in Java. PHP koda se mora nahajati med značkami <?php in ?>, tako da interpreter ve, kateri del kode je potrebno prevesti. Za razvoj aplikacije v ogrodju Laravel je bila uporabljena PHP verzija 5.5.12, vendar je trenutno v razvoju že verzija 7.1.0.

5.1.2 HTML

HTML (angl. HyperText Markup Language) je označevalni jezik, ki se uporablja pri izdelovanju spletnih straneh. Velja za eno od ključnih tehnologij za razvoj spletnih strani in uporabniške vmesnike. Jedro jezika HTML je predstaviti pogosto uporabljene elemente spletnih strani z uporabo značk (angl. tag). Nekaj primerov takih značk:

- <p>, značka za odstavek
- <a>, značka za hiperpovezavo
- <h1>, značka za naslov

Vsako definirano značko oziroma element je potrebno zaključiti z isto značko, ki vsebuje poševnico kot predpono (<html></html>). HTML dokumente je možno odpreti v spletnih brskalnih, saj ti pretvorijo HTML značke v vidne elemente na spletni strani. Poleg osnovnih elementov besedila, kot so predstavljeni zgoraj, so na voljo še značke za vizualne elemente in vključitev skript CSS in JavaScript. Slednje se uporabi za oblikovanje in razvoj interaktivne spletne strani. Čeprav je HTML star že več kot dve desetletji, je

še vedno v uporabi in ga trenutno vzdržuje organizacija World Wide Web Consortium (W3C).

5.1.3 CSS

CSS (angl. Cascading Style Sheets) oziroma kaskadne stilske predloge opišejo način prikaza in obliko HTML dokumentov na spletnih straneh. Z uporabo CSS datotek ločimo predstavitev HTML elementov od vsebine strani, kar je dobro, zato ker HTML ni bil namenjen uporabi značk za oblikovanje spletnih strani. S to ločitvijo je bila omogočena enostavnejša uporaba in nadzor nad HTML dokumenti. CSS datoteke prav tako omogočajo fleksibilnost in večkratno uporabo istih stilov za različne HTML dokumente. Spodaj je prikazan primer CSS datoteke.

```
td {
  text-align: right;
  width: 50px;
}
```

Za razvoj aplikacije je bilo uporabljeno HTML, CSS in JavaScript ogrodje Bootstrap, s katerim je omogočeno lažje oblikovanje in izdelovanje odzivnih spletnih strani.

5.1.4 Javascript in jQuery

JavaScript je dinamičen programski jezik, ki je poleg tehnologij HTML in CSS jedro za razvoj spletnih aplikacij in se uporablja za izboljšanje interaktivnosti spletnih straneh. Kljub podobnosti imena, sintakse in standardnih knjižnic se programska jezika JavaScript in Java močno razlikujeta po zasnovi. Primarno se JavaScript izvede na strani odjemalca, vendar popularnost narašča tudi platformam in ogrodjem, ki JavaScript kodo izvedejo na strani strežnika.

jQuery je najbolj priljubljena JavaScript knjižnica, ki poenostavi programiranje z JavaScript in omogoča hitrejši razvoj strani za odjemalca. jQuery močno pomaga pri obdelavi HTML dokumentov, spreminjanju CSS lastnosti, razne animacije in efekti, AJAX klici in ostale koristne stvari. Preprost primer uporabe knjižnice jQuery je viden spodaj.

```
$('#searchText').focusout(function() {
  $('#searchText').animate({width: 'toggle'}, 200);
  $('#searchText').val('');
  $('#searchButton').css('pointer-events', '');
});
```

5.1.5 AJAX

AJAX oziroma asinhroni JavaScript in XML je nabor spletnih tehnik in tehnologij, ki omogočajo asinhrono pridobivanje podatkov na strani odjemalca. To pomeni, da lahko spletna aplikacija pošilja ali prejema podatke v ozadju, brez poseganja v spletno stran. AJAX sam po sebi ni tehnologija, vendar je AJAX klice možno opraviti s tehnologijo JavaScript ali jQuery. Čeprav je v imenu omenjen XML, uporaba tega ni nujna, saj se dandanes večinoma uporablja JSON zapis podatkov.

5.1.6 SQL

Strukturirani povpraševalni jezik (angl. Structured Query Language) oziroma SQL je povpraševalni jezik, namenjen dostopu in obdelavi podatkov v podatkovni bazi. Štiri osnovni stavki v jeziku SQL so:

- SELECT, s katerim dostopamo do podatkov
- INSERT, s katerim vnesemo nov zapis v podatkovno bazo
- UPDATE, s katerim posodobimo obstoječ zapis

- DELETE, s katerim izbrišemo obstoječ zapis

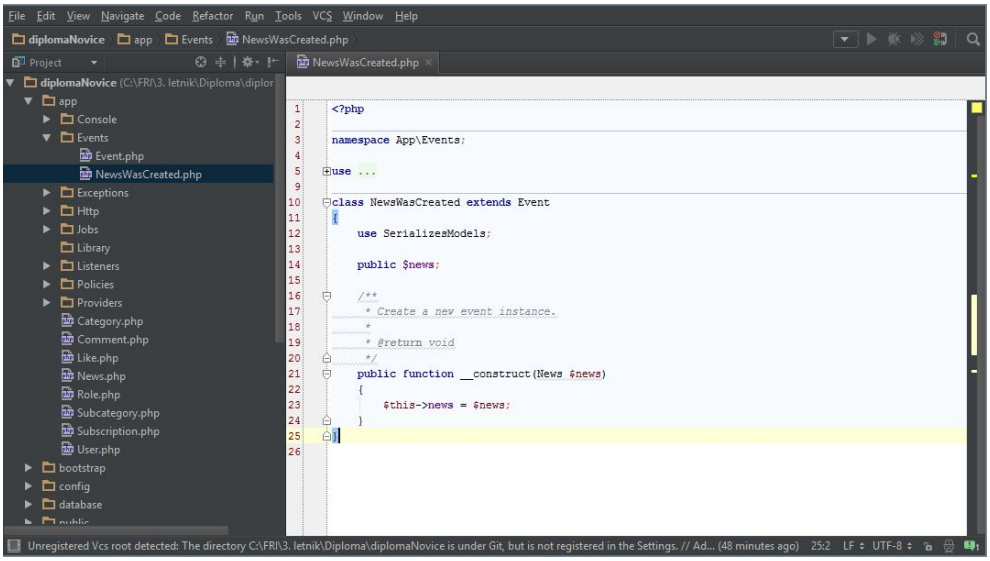
Poleg teh štirih osnovnih stavkov poznamo še stavke za ustvarjanje podatkovne baze, ustvarjanje nove tabele in še mnoge uporabne stavke za uporabo kompleksnih povpraševanj. Za podatkovno bazo smo uporabili sistem za upravljanje s podatkovnimi bazami (SUPB) MySQL, ki za delovanje uporablja zgoraj omenjeni povpraševalni jezik SQL.

5.2 Opis uporabljenih orodij

Preden se je pričelo z razvojem spletne aplikacije, je bilo potrebno postaviti razvojno okolje (angl. IDE – Integrated Development Environment), lokalni spletni strežnik (WAMP) ter sistem za upravljanje z izvorno kodo (Git). Uporabljeno je bilo še nekaj orodij, katere bomo opisali v nadaljnjem.

5.2.1 PhpStorm

JetBrains PhpStorm [13] je integrirano razvojno okolje, ki je namenjeno razvoju aplikacij v programskem jeziku PHP. To razvojno okolje nam ponuja urejevalnik izvorne kode za PHP, HTML in JavaScript z vgrajeno analizo izvorne kode, ki preverja pravilnost sintakse. Prav tako je na voljo pregled možnih napak in refaktoriranje kode. Dodatna funkcionalnost tega urejevalnika je možnost samodejnega dokončevanja (angl. autocomplete), katera je uporabna skozi celoten razvoj. Poleg urejevalnika izvorne kode PhpStorm ponuja še razne podpore za razhroščevanje, integracijo s sistemom za upravljanje z izvorno kodo itd. Žal PhpStorm ni na voljo brezplačno, vendar lahko študenti pridobijo licenco za izobraževalne namene. Na sliki 5.1 se vidi to razvojno okolje.



Slika 5.1: Prikaz integriranega razvojnega okolja PhpStorm.

5.2.2 Git

Git je eden izmed najbolj priljubljenih in uporabljenih sistemov za upravljanje z izvorno kodo. Tak sistem omogoča lažji razvoj v ekipi, saj se izvorna koda nahaja na strežnikih (GitHub, Gitlab, Bitbucket, ...). Na teh strežnikih je možen pregled nad celo izvorno kodo in tudi pregled med različnimi verzijami. Iz takega strežnika si vsak član ekipe lahko pridobi svojo lokalno različico kode, katero lahko spremenijo in nato oddajo nazaj na strežnik, kjer je spremenjena koda na voljo vsem ostalim članom. Git lahko uporablja tudi samo en razvijalec, saj ponuja tudi izdelovanje varnostnih kopij. Za gostovanje našega Git repozitorija smo izbrali spletno storitev GitHub in je ena izmed največjih spletnih strežnikov za gostovanje Git repozitorijev z obilico dodatnih orodij in veliko skupnostjo.

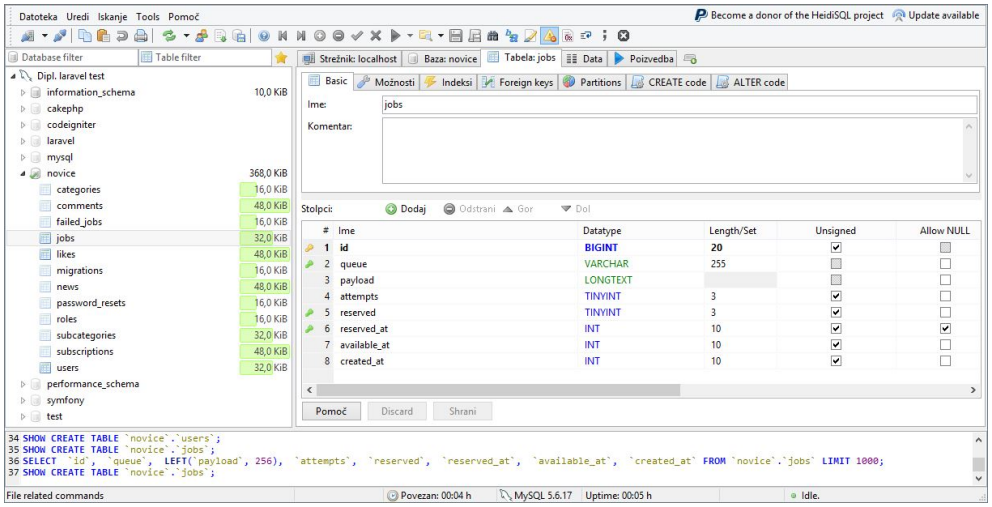
5.2.3 WampServer

WampServer je nabor odprtokodne programske opreme za operacijski sistem Windows, ki omogoča razvoj dinamičnih spletnih strani in spletnih aplikacij.

WAMP je akronim za Windows operacijski sistem, Apache spletni strežnik, MySQL sistem za upravljanje podatkovnih baz in PHP programski jezik. To orodje je potrebno za lokalno razvijanje spletne aplikacije.

5.2.4 HeidiSQL

HeidiSQL je odprtokodno orodje za dostop do lokalnih ali oddaljenih MySQL strežnikov. Uporaba tega orodja je dobro nadomestilo spletnega orodja phpMyAdmin. S HeidiSQL imamo pregled nad izbrano podatkovno bazo, prav tako pa lahko dodajamo, spreminjamo ali brišemo posamezne zapise. Poleg standardnih operacij nad podatki je možno še spreminjanje podatkovne baze ali posameznih tabel, ter izvajanje samostojnih SQL poizvedb. To je še posebej uporabno za testiranje pravilnosti SQL poizvedb, preden jih uporabimo v izvorni kodi projekta. Prikaz orodja lahko vidimo na sliki 5.2.



Slika 5.2: Prikaz orodja HeidiSQL.

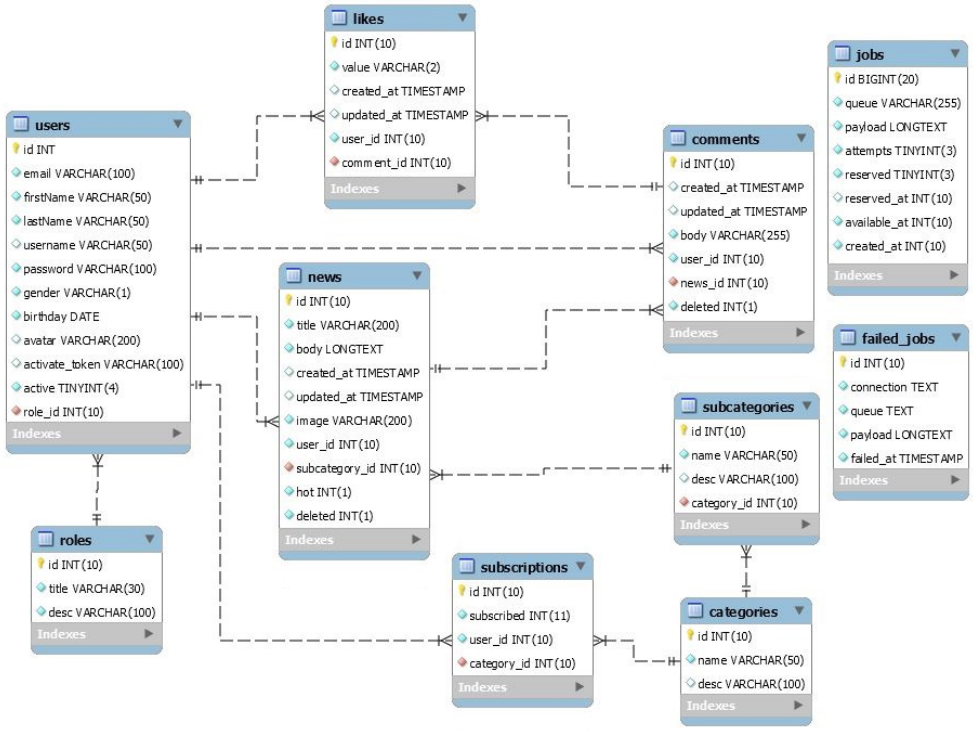
5.2.5 MySQL Workbench

MySQL Workbench je orodje za vizualno oblikovanje podatkovnih baz in vsebuje orodja za nastavitev strežnika in administracijo uporabnikov. Orodje

je bilo uporabljeno za razvoj podatkovne baze za spletno aplikacijo. Čeprav je možno to podatkovno bazo izvoziti v SQL datoteko, katero lahko izvršimo na podatkovnem strežniku, se za to nismo odločili, saj grajenje podatkovne baze v ogrodju Laravel poteka malce drugače.

5.3 Podatkovna baza

Kot smo že omenili v prejšnjih poglavjih, je za delujočo spletno aplikacijo pomembna podatkovna baza in je eden izmed pomembnejših delov, katerega je potrebno načrtovati še preden se lotimo izdelave aplikacije. Kljub temu da podatkovno bazo načrtujemo preden začnemo z razvojem, se med tem pogosto pripeti, da je podatkovni bazi potrebno nekaj spremeniti ali dodati. V nadaljnjem bomo opisali pomembne dele uporabljene podatkovne baze, katero lahko vidimo na sliki 5.3.



Slika 5.3: Prikaz podatkovne baze.

5.3.1 Tabela uporabnikov

V tabeli "users" (slo. uporabniki) se nahajajo vsi uporabniki spletne aplikacije. Vloge so dodeljene ob prijavi, katere se dobijo iz tabele "roles" (slo. vloge). Spletni uporabniki imajo možnost ogleda novic, komentiranje le-teh, glasovanje ostalih komentarjev in naročanje na novice. Uporabniki, katerih vloga je avtor, lahko ustvarjajo, posodabljajo in brišejo novice.

5.3.2 Tabela novic

Tabelo "news" (slo. novice) v osnovi sestavljajo naslov, telo novice, podkategorija, slika in identifikator uporabnika oziroma v tem primeru avtorja. Avtor ob kreiranju nove novice lahko označi novico kot "vročo", kar pomeni da bo novica objavljena na naslovni strani. Brisanje zapisov v podatkovnih bazah je težje in lahko postane zapleteno, zato zapisov ne izbrišemo iz baze, ampak jih označimo kot izbrisane (angl. "soft delete").

5.3.3 Tabela komentarjev

Za pregled komentarjev je uporabljena tabela "comments" (slo. komentarji), v katero lahko dodajajo samo spletni uporabniki. Brisanje neprimernih komentarjev je omogočeno administratorjem in lastnemu uporabniku. To je prav tako, kot pri brisanju novic implementirano na način "soft delete". Uporabniki, ki niso avtor komentarja, lahko glasujejo za katerikoli komentar le enkrat, imajo pa možnost ponovnega glasovanja v primeru, da je uporabnik storil napako in glasoval napačno.

5.3.4 Tabela rubrik

V tabeli "categories" (slo. kategorije) se nahajajo že vnaprej določene rubrike spletne strani. Ta tabela je tesno povezana s tabelo podrubrik, kjer so opisane povezave rubrik s podrubrikami, kot na primer rubrika "politika" vsebuje podrubriki "doma" in "po svetu". Te podrubrike so pomembne za ustvarjanje

novic, saj lahko tako uvrstimo novice po pravilnih rubrikah. Tako se lahko spletni uporabnik naroča, saj se naroča na izbrane rubrike in tako dobi novice teh podrubrik.

5.3.5 Tabela poslov

Tabeli "jobs" in "failed_jobs" sta samostojni in sta uporabljeni za pravilno delovanje čakalnih vrst za ogrodje Laravel. V tabelo "jobs" se avtomatično vnesejo posli, ki se kasneje v ozadju izvedejo samodejno.

5.4 Opis pomembnih rešitev

V prejšnjem poglavju smo omenili nekaj ključnih funkcionalnosti spletne aplikacije. V temu podpoglavju bomo te funkcionalnosti opisali na tehničnem nivoju.

5.4.1 Avtentikacija in avtorizacija

Kot smo že omenili, je v ogrodju Laravel možno postaviti preprost skelet za omogočanje registracije in prijave v sistem. To se zlahka doseže z uporabo konzolne vrstice, v katero vpišemo ukaz *php artisan make:auth*.

S tem se nam avtomatično ustvari krmilnik, ki skrbi za registracijo in prijave, ter pogledi, potrebni za pravilno registracijo in pravilno prijavo v sistem. Poleg tega se ustvari še krmilnik za ponastavitev gesel, kateri prav tako deluje "out of the box". Krmilnik, ki skrbi za avtentikacijo v sistem uporablja dva razreda, ki vsebujeta potrebne metode za avtentikacijo. V teh razredih se nahaja celotna logika prijavljanja v sistem, katero je bilo potrebno prilagoditi naši aplikaciji. Primer take prilagoditve oziroma spremembe je pri prijavi uporabnika, kjer je pred prijavo potrebno preveriti ali je uporabnik aktiviran. To lahko vidimo na naslednjem izseku kode.

```
//ce uporabnik ni aktiviran se ne more prijaviti
$user = User::where('email', $credentials['email'])->first();
if(isset($user)) {
    if(!Hash::check($credentials['password'], $user->password))
    {
        return $this->sendFailedLoginResponse($request);
    }
    if($user->active === 0) {
        return $this->sendNonActiveResponse($request);
    }
}
```

Preden se uporabnik lahko prijavi, se mora v aplikacijo registrirati, za kar skrbi logika v razredu RegistersUsers, kjer se nahaja nekaj preprostih metod, ki skrbijo za pravilno registracijo. Prav tako je bila tu potrebna manjša prilagoditev kode. Dodana je bila funkcionalnost pošiljanja e-pošte za aktivacijo spletnega računa. Seveda je pred izvedbo logike nad podatki potrebno preveriti pravilnost vnešenih podatkov. S postavljanjem skeleta se ta funkcija avtomatično izdela, vendar jo je potrebno prilagoditi lastnim potrebam. Validacijo podatkov za registracijo lahko vidimo na naslednjem delu kode.

```
protected function validator(array $data)
{
    return Validator::make($data, [
        'firstName' => 'required|max:50',
        'lastName' => 'required|max:50',
        'username' => 'required|max:50',
        'gender' => 'required',
        'birthday' => 'required|before:1.1.2005',
        'email' => 'required|email|max:100|unique:users',
        'password' => 'required|min:6|confirmed',
    ]);
}
```

Kot se lahko opazi, se potrjevanje podatkov v ogrodju Laravel izvede s pomočjo tabel, v katerih lahko verižimo različna potrjevalna pravila, kot na primer obveznost polja in omejeno število vpisanih znakov. Omogočeno je tudi dodajanje lastnih validacijskih pravil, vendar tega v tej aplikaciji ni bilo potrebno opraviti.

V spletnih aplikacijah je potrebno imeti postavljeno avtorizacijo sistema, tako da neavtorizirani uporabniki nimajo vstopa na strani, kjer njim ni dovoljeno. To avtorizacijo je potrebno implementirati posebej, saj se ne postavi samodejno z uporabo avtentikacijskega skeleta. Avtentikacija se v ogrodju Laravel izvede s pomočjo vmesnih mehanizmov (angl. middleware), kateri se izvedejo pred vstopom v krmilnike in tako lahko prepovemo vstop. Tak vmesni mehanizem ustvarimo s pomočjo konzolne vrstice, kar ustvari razred Middleware, v katerega podamo lastno metodo, v tem primeru za avtorizacijo sistema. V naslednjem primeru opazimo avtorizacijo spletnih uporabnikov, v aplikaciji pa se uporablja še avtorizacija administratorja in avtorjev.

```
public function handle($request, Closure $next)
{
    //vstop uporabnikom
    if(!$request->user()->isUser()) {
        return redirect(Config::get('paths.PATH_ROOT'))
        ->withErrors(['error' => trans('errors.unathourized')]);
    }
    return $next($request);
}
```

Ta vmesni mehanizem moramo nato zapisati v datoteko Kernel.php, kjer so zapisani vsi vmesni mehanizmi. Prikazan je primer za prejšnje opisan middleware.

```
protected $routeMiddleware = [
    'user'  => \App\Http\Middleware\UserMiddleware::class,
];
```

Vmesni mehanizem “user” se lahko zdaj uporablja skozi celotno aplikacijo in se nastavi posameznim krmilnikom v konstruktni metodi.

```
public function __construct() {
    $this->middleware(['auth', 'user']);
}
```

Seveda je ponekod potrebna avtorizacija posameznega dela na spletni strani. Za to se definira pravila v razred oziroma storitev, ki skrbi za avtorizacijo sistema. V našem primeru uporabljamo več takih pravil, vendar je na naslednjem izseku razvidno samo eno.

```
public function boot(GateContract $gate)
{
    $this->registerPolicies($gate);

    $gate->define('isUser', function($user){
        return $user->isUser();
    });
}
```

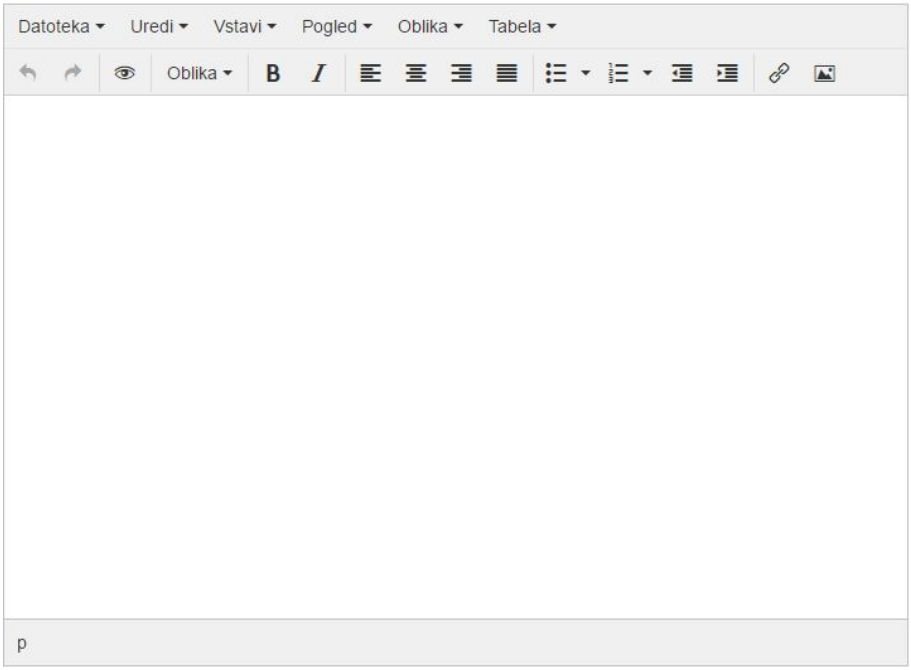
Taka pravila lahko sedaj uporabimo v metodah ali pa v pogledih s pomočjo razčlenjevalnika predlog - Blade. To storimo s pomočjo niza @can, ki je prikazan na naslednjemu primeru.

```
@can('isUser')
    <button id="postComments" type="button">
        {{trans('views\individualNews.postComment')}}
    </button>
@endcan
```

5.4.2 Ustvarjanje novic

Rdeča nit te spletne aplikacije je ogled in ustvarjanje novic. Zato je bilo potrebno izdelati način intuitivnega pisanja novic in dodajanje in upravljanje slik. Za urejevalnik besedil smo uporabili odprtokoden JavaScript urejevalnik TinyMCE. Vsebuje ogromno uporabnih orodij in funkcij, ki pomagajo ustvariti karseda intuitiven urejevalnik.

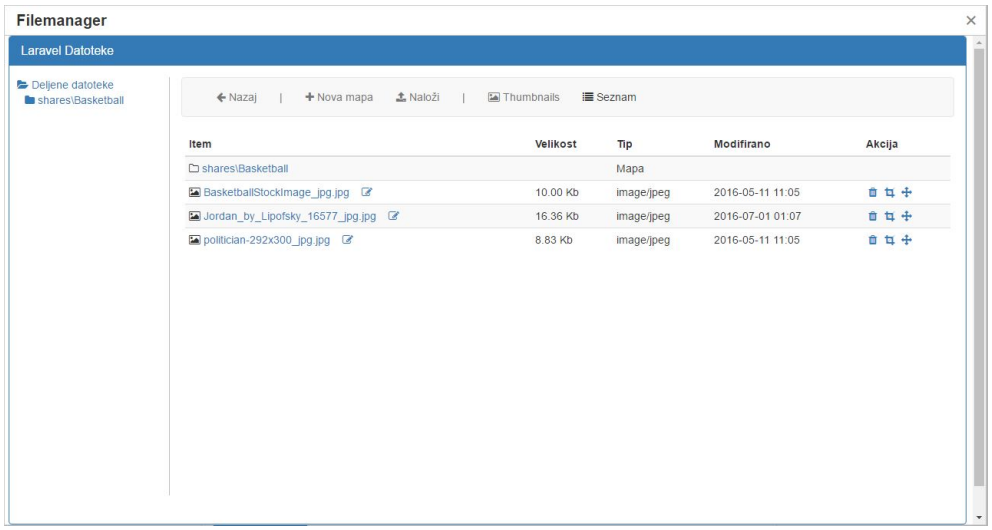
Čeprav je osnovna verzija urejevalnika TinyMCE brezplačna, imajo na voljo tudi plačljive verzije, v katerih ponujajo dodatna zahtevnejša orodja in podporo. Na sliki 5.4 lahko vidimo urejevalnik besedila TinyMCE.



Slika 5.4: Prikaz urejevalnika besedil TinyMCE.

Pri ustvarjanju novic je pomembno, da imamo dostop do upravljanja s slikovnimi elementi. To je bilo potrebno integrirati z urejevalnikom besedila TinyMCE, za kar smo uporabili odprtokodno komponento za ogrodje Laravel – Unisharpov upravljalec datotek. Za to komponento smo se odločili zato, ker je integracija z urejevalnikom TinyMCE enostavna. Če želimo dostopati

do tega upravljalca datotek, je potrebno klikniti na gumb za vstavitev slike. S pomočjo te komponente lahko zdaj avtor ali administrator nalaga slike, jih preimenuje, preoblikuje in briše. Ko so zaželjene slike naložene preko vmesnika, jih lahko avtor vključi v svoje novice. Upravljalec datotek lahko vidimo na sliki 5.5.



Slika 5.5: Prikaz upravljalca datotek.

5.4.3 Naročanje na novic

Naročanje na novice je ena ključnih funkcionalnosti spletne aplikacije in je bila implementirana s pomočjo čakalnih vrst, ki jih ogrodje Laravel ponuja. Vsak spletni uporabnik se lahko naroči na izbrano rubriko novic. Če uporabnik to stori, se mu vsakič, ko je ustvarjena nova novica pod to rubriko, pošlje novica na njegov e-poštni naslov. Zato je bilo najprej potrebno implementirati pošiljanje e-pošte s pomočjo razreda *Job*, katerega lahko nato vključimo v čakalno vrsto. V ta razred pošljemo objekt uporabnika, iz katerega pridobimo podatke potrebne za pošiljanje elektronske pošte. Telo elektronskega sporočila pridobimo iz pogleda, katerega napolnimo s podatki, kot ostale poglede v ogrodju Laravel.

Za pravilno pošiljanje teh elektronskih sporočil je bilo potrebno še implementirati dogodek in poslušalca, s katerima je moč sprožiti pošiljanje sporočil, ko se ustvari nova novica. Dogodek je preprost razred, v katerem obstaja samo objekt ustvarjene novice. Nato se izdela poslušalca, ki izvede vso logiko te funkcionalnosti. V temu poslušalcu pregledamo, pod katero rubriko je bila dodana novica, s katero lahko zatem poiščemo vse naročene uporabnike. Ko imamo na voljo vse naročnike, lahko uporabimo razred *Job* za pošiljanje elektronskih sporočil, katerega smo prej omenili. Na koncu je potrebno še registrirati dogodek s poslušalcem, tako da ogrodje ve, katerega poslušalca mora sprožiti ob določenem dogodku. Na naslednjem izseku kode lahko vidimo poslušalca, ki skrbi za pošiljanje elektronske pošte vsem naročnikom.

```
class SubscriberEmailSend implements ShouldQueue
{
    use DispatchesJobs;

    public function handle(NewsWasCreated $event)
    {
        $news = $event->news;
        $catId = News::getCatForNews($news->id)->id;
        $subs = Subscription::where('category_id', '=', $catId)
                  ->get();

        foreach($subs as $sub) {
            $user = $sub->user;
            if($sub->subscribed == 1) {
                $this->dispatch(new SendSubscriptionMail($user,
                $news));
            }
        }
    }
}
```

Ta poslušalec skrbi za uvrščanje del v čakalno vrsto, vendar je v ozadju potrebno omogočiti poslušalca čakalnih vrst, kar storimo v ukazni vrstici z

ukazom *php artisan queue:listen*. S tem zdaj v ozadju teče poslušalec čakalnih vrst, ki čaka na razna dela kot na primer sprožanje dogodkov in pošiljanje e-pošt.

5.4.4 Podatki o vremenu

Na naslovni strani lahko ob strani vidimo podatke o vremenu na lokaciji, kjer se nahaja uporabnik. Prikazani so podatki o trenutnem vremenu in vremenu za naslednje štiri dneve. Na sliki 5.6 lahko vidimo primer stranske vrstice z vremenom, kjer je trenutna lokacija Ljubljana.

Slika 5.6: Prikaz podatkov o vremenu.

Celotna funkcionalnost deluje v programskem jeziku JavaScript in deluje s pomočjo uporabnih API strani, s katerih pridobimo razne podatke asinhrono z AJAX klici. V naslednjem izseku kode vidimo poenostavljene AJAX klice z uporabo jQueryja.

```
$(document).ready(function() {
    $.when( //pridobi geo podatke
        $.get(geoUrl, function(data) {
                setGeoData(data);
```

```
        })
    ).then(function(){
        if(minutesPassed(10)) {
            $.when( //pridobi podatke o vremenu
                $.get(fullWeatherCurrUrl, function (data) {
                    setCurrData(data);
                }),
                $.get(fullWeatherUrl, function (data) {
                    setFourData(data);
                })
            ).then(function () { //nastavi podatke o vremenu
                    setCurrWeather();
                    setFourWeather();
            })
        }
        else {
            setCurrWeather();
            setFourWeather();
        }
    });
});
```

Trenutno lokacijo dobimo z uporabo spletne strani ip-api (dostopno na http://ip-api.com/json), odkoder pridobimo geolokacijske podatke, ki so potrebni za informacije o trenutni lokaciji. Te podatke dobimo v obliki JSON, zato je treba pomembne podatke izluščiti iz tega zapisa. Podatke, katere potrebujemo so ime mesta, zemljepisna dolžina in širina. S temi tremi podatki lahko opravimo dodaten AJAX klic za podatke o vremenu.

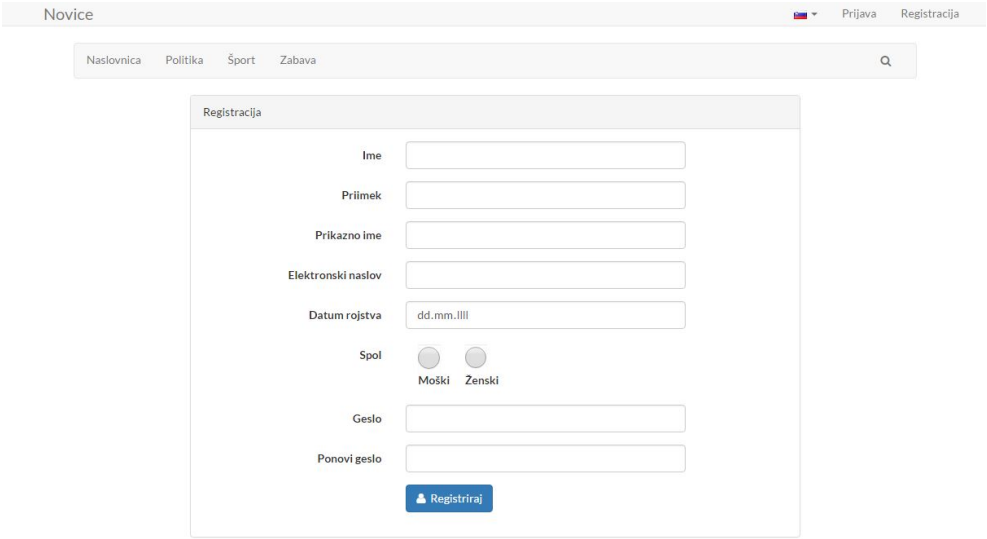
S pridobljenimi geolokacijskimi podatki lahko sedaj pridobimo podatke o trenutnem vremenu in vremenski napovedi s storitvijo na spletni strani OpenWeather (dostopno na http://openweathermap.org/api). Ta storitev je plačljiva mesečno, vendar je na voljo tudi brezplačna verzija, v kateri je omogočeno bistveno manj klicev na minuto. Zato je bilo potrebno klice na njihovo stran omejiti enkrat na deset minut. Opraviti je treba dva AJAX klica: klic za podatke o trenutnem vremenu in klic za podatke o napovedi naslednjih štiri dni. Pridobljene podatke smo nato shranili v sejo brskalnika,

ki nima datum poteka (localStorage).

Kasneje te podatke uporabimo pri izrisu v stransko vrstico. Za izris stanja vremena smo uporabili vremenske ikone z uporabo ogrodja Bootstrap in komponento vremenskih ikon (Weather Icons).

5.5 Potek aplikacije

Namen aplikacije je ogled novic, kar je na voljo tudi neprijavljenim uporabnikom. Vendar če se uporabniki želijo naročati na novice in jih komentirati, se morajo prej registrirati. To lahko preprosto naredijo s klikom na gumb za registracijo, kjer vpišejo nekaj pomembnih podatkov, kot so na primer e-poštni naslov in geslo. Poleg teh dveh podatkov je potrebno vpisati še nekaj podatkov, ki so pomembni za komentarje in pošiljanje elektronske pošte. Registracijo uporabnika lahko vidimo na sliki 5.7.

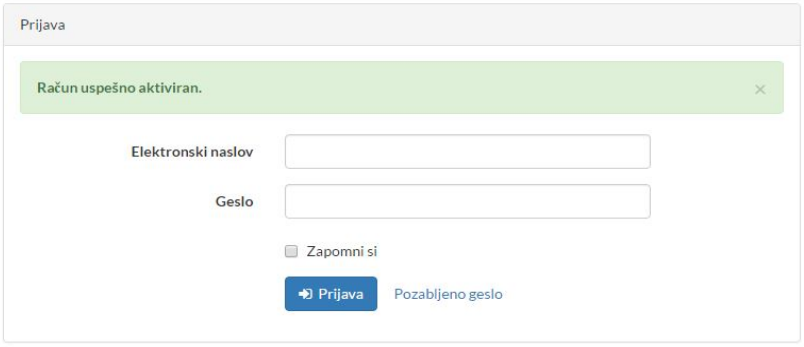


Slika 5.7: Registracija novega uporabnika.

Ko je registracija uspešna, se na podani e-poštni naslov pošlje aktivacijska povezava za aktiviranje novo ustvarjenega računa. Če ta račun ni aktiviran,

prijava v spletno aplikacijo ni možna. V primeru da uporabnik ne prejme e-pošte z aktivacijsko povezavo, obstaja možnost za ponovno pošiljanje te e-pošte.

Ko se odpre povezavo za aktivacijo, se račun aktivira in je prijava omogočena. To lahko vidimo na sliki 5.8.



Slika 5.8: Prijava uporabnika.

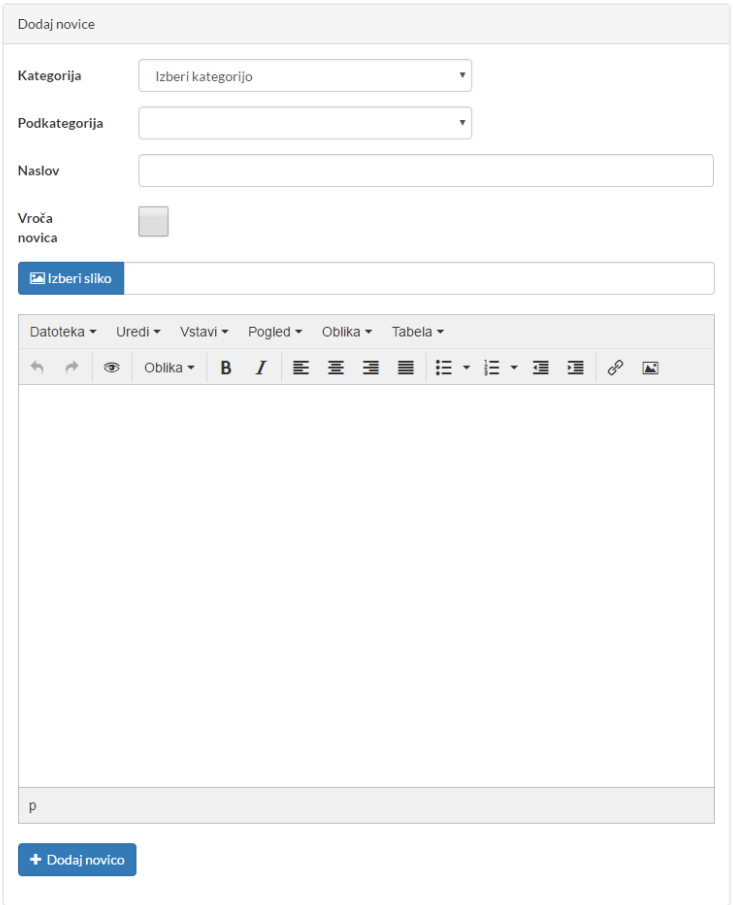
S klikom na izbirni gumb "Zapomni si" se ustvari piškotek, ki si zapomni uporabnikov elektronski naslov za en dan. V primeru da uporabnik pozabi svoje geslo, ima na prijavni strani na voljo povezavo za ponastavitev gesla. To se izvede podobno kot pri aktivaciji računa, le da se na elektronsko pošto pošlje povezava za ponastavitev gesla.

Registracija avtorjev poteka različno od registracije uporabnikov. Ker smo se odločili, da avtor ne more biti kdorkoli od uporabnikov, je dodajanje avtorjev naloga administratorja. Ta se mora najprej prijaviti v spletno aplikacijo in izbrati opcijo za dodajanje novega avtorja, ki se nahaja v glavnem meniju prijavljenega uporabnika.

Pri registraciji novega avtorja, administrator ne nastavi gesla avtorju. To je bilo implementirano zaradi varnosti avtorjevega računa. Ko administrator konča z registracijo avtorja, se temu na elektronski naslov pošlje e-pošta z aktivacijsko povezavo. Ta deluje podobno kot pri aktivaciji spletnega upo-

rabnika, vendar si mora novi avtor še nastaviti geslo za prijavo.

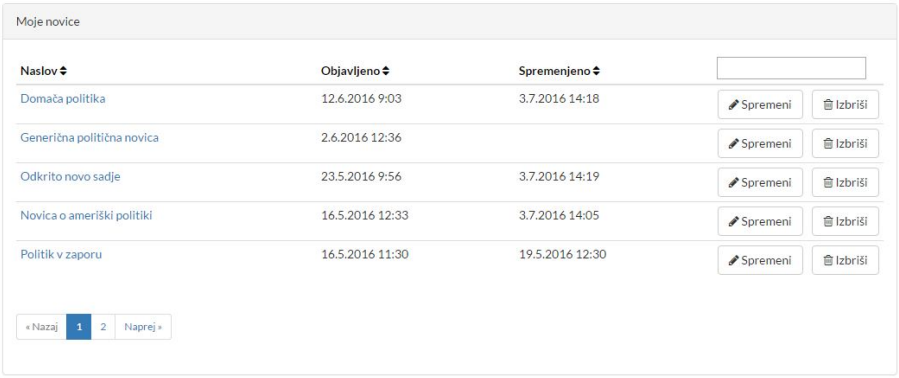
Ko je avtor registriran in prijavljen v sistem, ima možnost dodajanja novih novic v glavnem meniju. S klikom na izbiro "Dodaj novice", se odpre stran z obrazcem za ustvarjanje novic. Dodajanje nove novice se začne z izbiro rubrike, podrubrike, prikazne slike in možnost, da se novica pojavi na naslovni strani. Nato lahko avtor začne s pisanjem novice v urejevalniku besedila. Stran za ustvarjanje novic je razvidna na sliki 5.9.



Slika 5.9: Ustavarjanje nove novice.

Pregled nad vsemi novicami, ki jih je ustvaril avtor, je možen s klikom

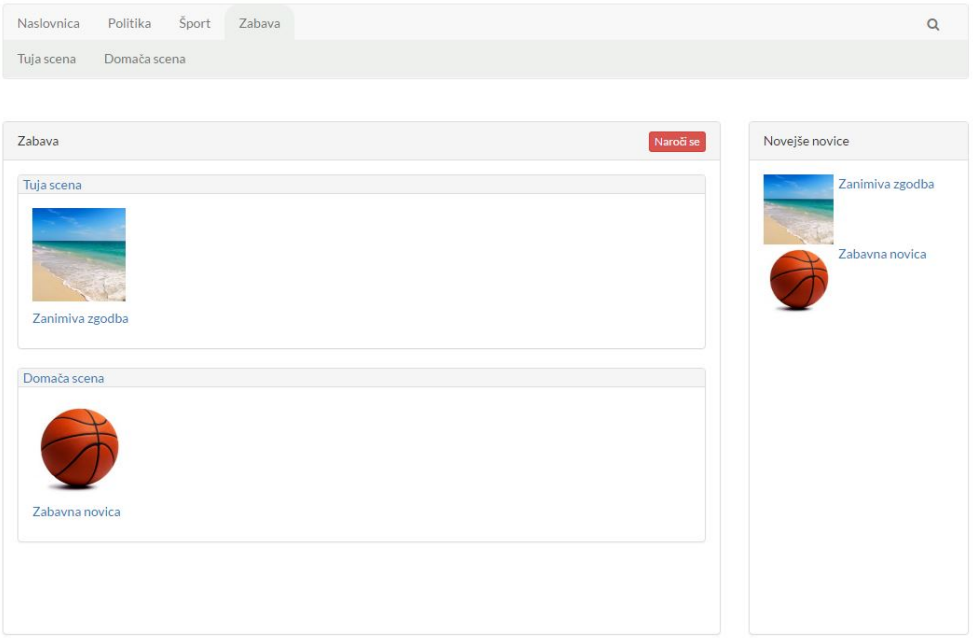
na gumb “Moje novice”. Tam so naštete vse avtorjeve novice s podatki o datumu objave in spremembe, povezava do novic. Prav tako je omogočeno razvrščanje oziroma sortiranje teh podatkov po naraščajočem ali padajočem vrstnem redu. Poleg sortiranja je na voljo še iskanje po novicah, kar lahko olajša iskanje zaželene novice. Pregled teh novic lahko vidimo na sliki 5.10.



Slika 5.10: Prikaz avtorjevih novic.

Za vsak zapis sta na volji še možnosti za urejanje in brisanje novice. S klikom na gumb za spreminjanje se avtorja preusmeri na enak obrazec, kot za ustvarjanje novice, le da so v tem primeru podatki že vstavljeni. S klikom na gumb “Izbriši” pa se pojavi okno, ki avtorja opozori na brisanje novice. Pregled novic je omogočen tudi administratorju, le da on vidi vse ustvarjene novice. Prav tako lahko ureja in briše novice.

Novo ustvarjene in spremenjene novice so tako na voljo za ogled obiskovalcem in spletnim uporabnikom, do katerih lahko dostopajo preko naslovne strani, strani rubrik ali pa preko iskanja novic. Na sliki 5.11 je prikazan pogled rubrike “Zabava”, kjer sta na voljo še dve podrubriki.

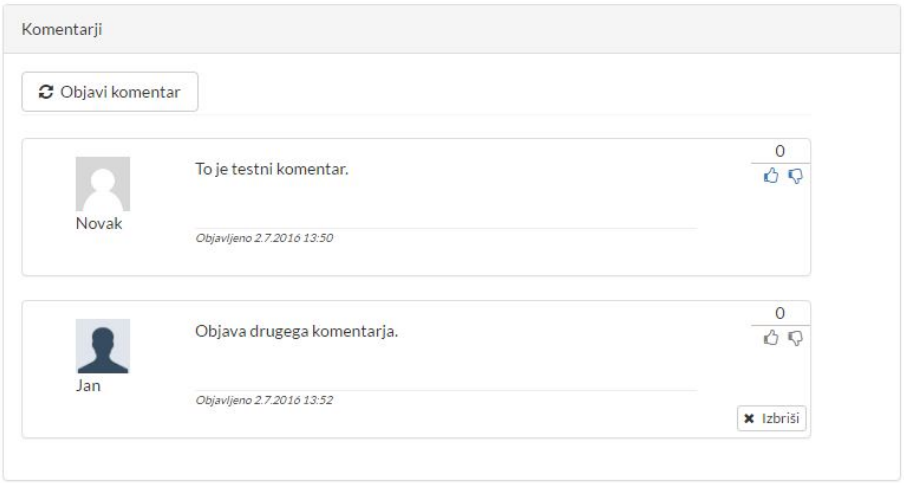


Slika 5.11: Prikaz rubrike “Zabava”.

V zgornjem desnem kotu okna za rubriko se opazi gumb za naročanje. S klikom na ta gumb se lahko prijavljeni uporabnik naroči na prejemanje novic te rubrike na svojo elektronsko pošto. Če se uporabnik želi odjaviti s te rubrike, to preprosto naredi s ponovnim klikom na gumb, ki se imenuje “Odjavi se”, če postavimo nad gumb miško. Poleg naročanja lahko opazimo še okno novejših novic, ki se nahaja na desni strani zaslona. Tam se nahajajo hitre povezave do najnovejših pet novic, ki so bile ustvarjene v tej rubriki (v našem primeru je to rubrika “Zabava”).

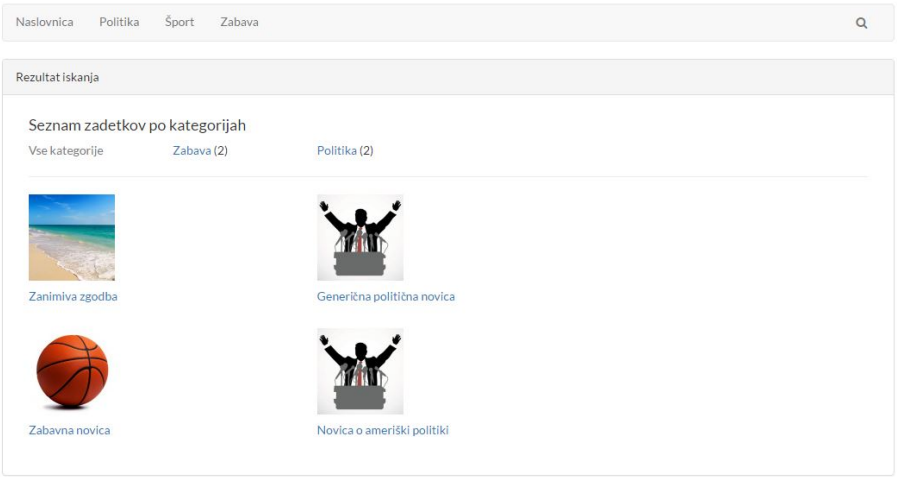
S klikom na eno od teh povezav se odpre stran, ki prikazuje izbrano novico in njene komentarje če te obstajajo.

Pod novico, se nahaja okno za vse komentarje, kjer lahko objavljajo in ocenjujejo komentarje samo registrirani uporabniki. Na voljo je tudi brisanje komentarjev, kar lahko storijo administrator in avtorji komentarjev. Pogled komentarjev lahko opazimo na sliki 5.12.



Slika 5.12: Prikaz okna za komentarje.

Iskanje novic lahko opravimo s klikom na povečevalno steklo, ki se nahaja desno na vrstici z navigacijskimi gumbi. Ob kliku na gumb za iskanje se začasno pojavi tekstovno polje, kamor zapišemo poljubni iskalni niz. S pritiskom na gumb "Enter" se odpre nova stran, kjer so prikazani zadetki iskanja. Če iskanje najde novice, se te lahko sortira po rubrikah. Primer za iskalni niz "novica" lahko vidimo na sliki 5.13.



Slika 5.13: Prikaz rezultata iskanja.

Poglavje 6

Sklepne ugotovitve

Tema diplomske naloge je bila primerjava najbolj priljubljenih PHP spletnih ogrodij, ki so trenutno na voljo. Za to primerjavo smo izbrali štiri naslednja ogrodja: Laravel, Symfony, CodeIgniter in CakePHP.

Primerjavo smo opravili na podlagi nekaj kriterijev, ki so po našem mnenju pomembni pri učenju, razvoju in uporabi teh ogrodij. Te kriteriji so bili: skupnost, dokumentacija, podpora in ključne funkcionalnosti ogrodja. Najbolj je pri skupnosti negativno izstopalo ogrodje CakePHP, saj je imelo kar pol manj priljubljenosti na spletni strani GitHub, kot ostala tri ogrodja. Ogrodji Symfony in CakePHP sta imeli bistveno bolj obsežno dokumentacijo v obliki elektronske knjige. Pri ogrodjih Laravel in Symfony je ponujena dolgotrajna podpora, kjer je ponujena večletna podpora pri razhroščevanju glavne verzije. Prav tako sta v številu in kvaliteti funkcionalnosti na vrhu ogrodji Laravel in Symfony, kjer ostalima dvema tekmeca primanjkuje kakšna pomembna oziroma uporabna funkcionalnost. Na koncu je bilo izbrano ogrodje Laravel zahvaljujoč svoji preprostosti, uporabnosti in hitro rastoči skupnosti.

Ko je bilo spletno ogrodje izbrano, smo pričeli z izdelavo spletne aplikacije, katera je bila spletni portal za ogled novic. S pomočjo tega spletnega portala lahko obiskovalci in uporabniki ogledujejo novice, katere izdelajo avtorji oziroma novinarji s strani portala. Z izdelavo tega portala je bila prikazana

zmogljivost, hitra naučljivost in nekaj prednosti spletnega ogrodja Laravel. Zanimiva prednost tega ogrodja je preprosta implementacija čakalnih vrst, katero smo uporabili za izvajanje del samostjno v ozadju aplikacije.

Prispevki tega diplomskega dela so lahko uporabni razvijalcem ali vodjam, ki se morajo soočiti z izbiro spletnega ogrodja za programski jezik PHP. Z izdelavo spletnega portala je mogoče videti enostavnost razvoja in uporabe zunanjih komponent, katere lahko bistveno zmanjšajo čas razvoja.

Pri izdelavi spletne aplikacije menimo, da bi se morala izboljšati oblika oziroma izgled spletnega portala. Poleg te izboljšave bi bila primerna še implementacija spletnega foruma, kjer bi lahko spletni uporabniki sodelovali v različnih razpravah.

Literatura

[1] M. Bean. *Laravel 5 Essentials*. Packt Publishing, Birmingham, United Kingdom, 2015, str. 3–9.

[2] The Symfony Book. [Online]. Dosegljivo: https://symfony.com/pdf/Symfony_book_3.1.pdf?v=4. [Dostopano 15. 6. 2016].

[3] CodeIgniter User Guide. [Online]. Dosegljivo: http://www.codeigniter.com/user_guide/. [Dostopano 16. 6. 2016].

[4] A. Bari, A. Syam. *CakePHP Application Development*, Packt Pub., Birmingham, United Kingdom, 2008.

[5] Laravel documentation. [Online]. Dosegljivo: https://laravel.com/docs/5.2. [Dostopano 14. 6. 2016].

[6] CakePHP Cookbook Documentation. [Online]. Dosegljivo: http://book.cakephp.org/3.0/_downloads/en/CakePHPCookbook.pdf. [Dostopano 17. 6. 2016].

Spletni viri

[7] Usage statistics and market share of PHP for websites. [Online]. Dosegljivo: https://w3techs.com/technologies/details/pl-php/all/all. [Dostopano 14. 7. 2016].

[8] 15 Best free PHP frameworks. [Online]. Dosegljivo: http://beebom.com/best-free-php-frameworks/. [Dostopano 30. 3. 2016].

[9] Symfony release process. [Online]. Dosegljivo: http://symfony.com/doc/current/contributing/community/releases.html#releases-lts. [Dostopano 15. 6. 2016].

[10] Symfony fundamentals. [Online]. Dosegljivo: http://symfony.com/doc/current/book/http_fundamentals.html. [Dostopano 15. 6. 2016].

[11] CodeIgniter application flow. [Online]. Dosegljivo: http://www.codeigniter.com/user_guide/overview/appflow.html. [Dostopano 16. 6. 2016].

[12] CakePHP at a Glance. [Online]. Dosegljivo: http://book.cakephp.org/3.0/en/intro.html. [Dostopano 17. 6. 2016].

[13] JetBrains PhpStorm. [Online]. Dosegljivo: https://www.jetbrains.com/phpstorm/. [Dostopano 13. 7. 2016].